

Juniper Networks NetScreen CLI Reference Guide

Version 5.0.0

Command Descriptions

Version 5.0.0

P/N 093-1352-000

Rev. A

Copyright Notice

Copyright © 2004 Juniper Networks, Inc. All rights reserved.

Juniper Networks, the Juniper Networks logo, NetScreen, NetScreen Technologies, GigaScreen, and the NetScreen logo are registered trademarks of Juniper Networks, Inc. NetScreen-5GT, NetScreen-5XP, NetScreen-5XT, NetScreen-25, NetScreen-50, NetScreen-100, NetScreen-204, NetScreen-208, NetScreen-500, NetScreen-5200, NetScreen-5400, NetScreen-Global PRO, NetScreen-Global PRO Express, NetScreen-Remote Security Client, NetScreen-Remote VPN Client, NetScreen-IDP 10, NetScreen-IDP 100, NetScreen-IDP 500, GigaScreen ASIC, GigaScreen-II ASIC, and NetScreen ScreenOS are trademarks of Juniper Networks, Inc. All other trademarks and registered trademarks are the property of their respective companies.

Information in this document is subject to change without notice.

No part of this document may be reproduced or transmitted in any form or by any means, electronic or mechanical, for any purpose, without receiving written permission from:

Juniper Networks, Inc.

ATTN: General Counsel

1194 N. Mathilda Ave. Sunnyvale, CA 94089

FCC Statement

The following information is for FCC compliance of Class A devices: This equipment has been tested and found to comply with the limits for a Class A digital device, pursuant to part 15 of the FCC rules. These limits are designed to provide reasonable protection against harmful interference when the equipment is operated in a commercial environment. The equipment generates, uses, and can radiate radio-frequency energy and, if not installed and used in accordance with the instruction manual, may cause harmful interference to radio communications. Operation of this equipment in a residential area is likely to cause harmful interference, in which case users will be required to correct the interference at their own expense.

The following information is for FCC compliance of Class B devices: The equipment described in this manual generates and may radiate radio-frequency energy. If it is not installed in accordance with NetScreen's installation instructions, it may cause interference with radio and television reception. This equipment has been tested and found to comply with the limits for a Class B digital device in accordance with the specifications in part 15 of the FCC rules. These specifications are designed to provide reasonable protection against such interference in a residential installation. However, there is no guarantee that interference will not occur in a particular installation.

If this equipment does cause harmful interference to radio or television reception, which can be determined by turning the equipment off and on, the user is encouraged to try to correct the interference by one or more of the following measures:

- Reorient or relocate the receiving antenna.
- Increase the separation between the equipment and receiver.
- Consult the dealer or an experienced radio/TV technician for help.
- Connect the equipment to an outlet on a circuit different from that to which the receiver is connected.

Caution: Changes or modifications to this product could void the user's warranty and authority to operate this device.

Disclaimer

THE SOFTWARE LICENSE AND LIMITED WARRANTY FOR THE ACCOMPANYING PRODUCT ARE SET FORTH IN THE INFORMATION PACKET THAT SHIPPED WITH THE PRODUCT AND ARE INCORPORATED HEREIN BY THIS REFERENCE. IF YOU ARE UNABLE TO LOCATE THE SOFTWARE LICENSE OR LIMITED WARRANTY, CONTACT YOUR NETSCREEN REPRESENTATIVE FOR A COPY.

Contents

Preface	vii	BGP Commands	61
Organization.....	vii	advertise-def-route.....	65
CLI Command Syntax Format.....	viii	aggregate	66
Dependency Delimiters	viii	always-compare-med	68
Availability of CLI Commands and Features	viii	as-number	69
Object Name Conventions.....	ix	as-path-access-list	70
CLI Variables.....	x	community-list	73
Variable Notation.....	x	confederation	77
Common CLI Variable Names	xi	enable	79
Juniper Networks NetScreen Publications	xiii	flap-damping.....	80
“active-user” through “common-criteria”	1	hold-time	81
active-user.....	2	keepalive	82
address.....	3	local-pref	83
admin	5	med.....	84
alarm	17	neighbor	86
alias	23	network.....	97
all	24	redistribute	99
arp.....	25	reflector	101
attack.....	28	reject-default-route	103
attack-db	35	retry-time	104
audible-alarm	36	synchronization	105
auth	38	chassis	106
auth-server	43	clock	108
av	52	common-criteria.....	110
		“config” through “interface”	111
		config	112
		console.....	115

counter	118	"ip" through "port-mode"	241
delete	121	ip	242
di	124	ip-classification	244
dip	137	ippool	245
dns	139	l2tp	246
domain	143	led	254
downgrade	144	lcd	255
Syntax	144	license-key	257
Keywords and Variables	144	log	260
Downgrades and NSM	145	mac	270
envar	146	mac-learn	271
event	148	memory	273
exit	153	mirror	276
failover	154	modem	277
file	157	nrtp	284
firewall	159	nsmgmt	286
flow	161	nsrp	292
gate	169	ntp	312
group	170	os	317
group-expression	174	OSPF Commands	319
hostname	177	advertise-def-route	323
ike	178	area	325
ike-cookie	204	auto-vlink	328
interface	206	config	329
		database	330
		enable	334
		hello-threshold	335

interface	336	interface	419
lsa-threshold	337	invalid-timer	420
neighbor	339	max-neighbor-count	421
redistribute	340	neighbors	422
reject-default-route	342	no-source-validation	423
rfc-1583	343	redistribute	424
routes-redistribute	344	reject-default-route	426
rules-redistribute	345	route-map	427
statistics	346	routes-redistribute	429
stub	347	rules-redistribute	430
summary-import	348	threshold-update	431
vlink	350	timer	432
vneighbor	354	trusted-neighbors	433
performance	355	update-timer	434
ping	357	update-threshold	435
pki	359	route	436
policy	376	sa	439
port-mode	395	sa-statistics	441
“pppoe” through “zone”	397	save	442
pppoe	398	scheduler	447
proxy-id	406	scp	450
reset	408	service	451
RIP Commands	409	session	455
advertise-def-route	412	sip	460
config	414	snmp	462
default-metric	415	socket	467
enable	416	ssh	468
flush-timer	417	ssl	476
garbage-list	418		

sys-clock.....	479	vpn	510
syslog.....	480	vpn-group	520
system	484	vpnmonitor	522
tech-support	485	vrouter	524
temperature-threshold.....	486	vsys.....	541
timer	487	webauth	544
trace-route	489	webtrends.....	546
traffic-shaping	491	xauth	548
url	493	zone	551
user.....	497	Interface Names	A-I
user-group.....	503	Zone Names	B-I
vip	508	Command Availability	C-I

Preface

The *Juniper Networks NetScreen CLI Reference Guide* describes the commands used to configure and manage a NetScreen device from a console interface. This manual is an ongoing publication, published with each NetScreen OS release.

ORGANIZATION

This guide consists of the following items:

- A *Table of Contents*, which lists the contents of all sections and appendices in this guide.
- Four *command sections*, each an alphabetized compendium of CLI command descriptions.

The command sections are as follows:

Section 1 describes CLI commands **active-user** through **common-criteria**.

Section 2 describes CLI commands **config** through **interface**.

Section 3 describes CLI commands **ip** through **port-mode**.

Section 4 describes CLI commands **pppoe** through **zone**.

- *Interface Names*, an appendix that lists and briefly describes NetScreen device interfaces.
- *Zone Names*, an appendix that lists and briefly describes zones.
- *Command Availability*, an appendix that lists the CLI commands and indicates which platforms support them.

CLI COMMAND SYNTAX FORMAT

Each CLI command description in this manual reveals some aspect of command syntax. This syntax may include options, switches, parameters, and other features. To illustrate syntax rules, some command descriptions use *dependency delimiters*. Such delimiters indicate which command features are mandatory, and in which contexts.

Dependency Delimiters

This manual uses the following conventions when presenting the syntax of a command line interface (CLI) command:

- Anything inside square brackets [] is optional.
- Anything inside braces { } is required.
- If there is more than one choice, each choice is separated by a pipe (|). For example,
`set interface { ethernet1 | ethernet2 | ethernet3 } manage`
means “set the management options for the ethernet1, ethernet2, or ethernet3 interface”.
- Variables appear in *italic*. For example:
`set admin user name1 password xyz`

When a CLI command appears within the context of a sentence, it is in **bold** (except for variables, which are always in *italic*). For example: “Use the **get system** command to display the serial number of a NetScreen device.”

Note: When typing a keyword, you only have to type enough letters to identify the word uniquely. For example, typing **set adm u joe j12fmt54** is enough to enter the command **set admin user joe j12fmt54**. Although you can use this shortcut when entering commands, all the commands documented here are presented in their entirety.

Availability of CLI Commands and Features

As you execute CLI commands using the syntax descriptions in this manual, you may find that certain commands and command features are unavailable for your NetScreen device model.

Because NetScreen devices treat unavailable command features as improper syntax, attempting to use such a feature usually generates the **unknown keyword** error message. When this message appears, confirm the feature's availability using the **?** switch. For example, the following commands list available options for the **set vpn** command:

```
ns-> set vpn ?  
ns-> set vpn vpn_name ?  
ns-> set vpn gateway gate_name ?
```

OBJECT NAME CONVENTIONS

ScreenOS employs the following conventions regarding the names of objects—such as addresses, admin users, auth servers, IKE gateways, virtual systems, VPN tunnels, and zones—defined in ScreenOS configurations.

- If a name string includes one or more spaces, the entire string must be enclosed within double quotes ("); for example, **set address trust "local LAN" 10.1.1.0/24**.
- NetScreen trims any spaces leading or trailing text within a set of double quotes; for example, **" local LAN "** becomes **"local LAN"**.
- NetScreen treats multiple consecutive spaces as a single space.
- Name strings are case sensitive, although many CLI key words are case insensitive. For example, **"local LAN"** is different from **"local lan"**.

ScreenOS supports the following character types:

- Single-byte character sets (SBCS) and multiple-byte character sets (MBCS). Examples of SBCS are ASCII, European, and Hebrew. Examples of MBCS—also referred to as double-byte character sets (DBCS)—are Chinese, Korean, and Japanese.

Note: A console connection only supports SBCS. The WebUI supports both SBCS and MBCS, depending on the character sets that your Web browser supports.

- ASCII characters from 32 (0x20 in hexadecimal) to 255 (0xff), except double quotes ("), which have special significance as an indicator of the beginning or end of a name string that includes spaces.

CLI VARIABLES

Most Juniper Networks NetScreen CLI commands have changeable parameters that affect the outcome of command execution. Juniper Networks NetScreen documentation represents these parameters as variables. Such variables may include names, identification numbers, IP addresses, subnet masks, numbers, dates, and other values.

Variable Notation

The variable notation used in this manual consists of italicized parameter identifiers. For example, the **set arp** command uses four identifiers, as shown here:

```
set arp
{
  ip_addr mac_addr interface
  age number |
  always-on-dest |
  no-cache
}
```

where

- *ip_addr* represents an IP address.
- *mac_addr* represents a MAC address.
- *interface* represents a physical or logical interface.
- *number* represents a numerical value.

Thus, the command might take the following form:

```
ns-> set arp 172.16.10.11 00e02c000080 ethernet2
```

where **172.16.10.11** is an IP address, **00e02c000080** is a MAC address, and **ethernet2** is a physical interface.

Common CLI Variable Names

The following is a list of CLI variable names used in NetScreen technical documentation.

<i>comm_name</i>	The community name of a host or other device.
<i>date</i>	A date value.
<i>dev_name</i>	A device name, as with flash card memory.
<i>dom_name</i>	A domain name, such as “acme” in www.acme.com .
<i>dst_addr</i>	A destination address.
<i>filename</i>	The name of a file.
<i>fqdn</i>	Fully-qualified domain name, such as www.acme.com .
<i>grp_name</i>	The name of a group, such as an address group or service group.
<i>interface</i>	A physical or logical interface.
<i>id_num</i>	An identification number.
<i>ip_addr</i>	An IPv4 address or IPv6 address.
<i>key_str</i>	A key, such as a session key, a private key, or a public key.
<i>key_hex</i>	A key expressed as a hexadecimal number.
<i>loc_str</i>	A location of a file or other resource.
<i>mac_addr</i>	A MAC address.
<i>mbr_name</i>	The name of a member in a group, such as an address group or a service group.
<i>mask</i>	A subnet mask, such as 255.255.255.224 or /24 .
<i>mcst_addr</i>	A multicast address.
<i>name_str</i>	The name of an item, such as “Marketing” for an address book entry.
<i>n</i>	A subnet mask (for an IPv4 subnet) or a prefix length (for an IPv6 subnet).
<i>number</i>	A numeric value, usually an integer, such as a threshold or a maximum.

<i>pol_num</i>	An integer that identifies a policy.
<i>port_num</i>	A number identifying a logical port.
<i>pref_len</i>	A number identifying the prefix length for an IPv6 address.
<i>pswd_str</i>	A password.
<i>ptcl_num</i>	A number uniquely identifying a protocol, such as TCP, IP, or UDP.
<i>serv_name</i>	The name of a server.
<i>shar_secret</i>	A shared secret value.
<i>spi_num</i>	A Security Parameters Index (SPI) number.
<i>src_addr</i>	A source address.
<i>string</i>	A character string, such as a comment.
<i>svc_name</i>	The name of a service, such as HTTP or MAIL.
<i>svc_num</i>	The ID number of a service, such as HTTP or MAIL.
<i>time</i>	A time value.
<i>tunn_str</i>	The name of a tunnel, such as an L2TP tunnel.
<i>url_str</i>	A URL, such as www.acme.com .
<i>usr_str</i>	A user name, usually an external entity such as a dialup user.
<i>vrouter</i>	A virtual router, such as trust-vr or untrust-vr.
<i>vsys</i>	The name of a vsys (virtual system).
<i>zone</i>	The name of a security zone.

Some commands contain multiple variables of the same type. The names of such variables might be numbered to identify each individually. For example, the **set dip** command contains two *id_num* variables, each numbered for easy identification:

```
set dip group id_num1 [ member id_num2 ]
```

JUNIPER NETWORKS NETSCREEN PUBLICATIONS

To obtain technical documentation for any Juniper Networks NetScreen product, visit www.juniper.net/techpubs/.

To obtain the latest software version, visit: www.juniper.net/support/. After logging in, select the Download Software option, and then follow the displayed instructions. (You must be a registered user to download Netscreen software.)

If you find any errors or omissions in the following content, please contact us at the e-mail address below:

techpubs@netscreen.com

“active-user” through “common-criteria”

This section lists and describes NetScreen Command Line Interface (CLI) commands **active-user** through **clock**.

Note: As you execute CLI commands using the syntax descriptions in this section, you may find that certain commands and command features are unavailable on your NetScreen platform. A good example is the **vsys** command, which is available on a NetScreen-500 device, but not on a NetScreen-5XP device. Similarly, some command options are unavailable on certain platforms, as with the **df-bit** option of the **vpn** command. This option is available on a NetScreen-500, but not on a NetScreen-5XP.

active-user

Description: Use the **active-user** command to display information for all users that initiated a service request through the NetScreen device. The displayed information includes the IP address of each user, and the number of sessions (incoming and outgoing) currently active for the user.

Note: *The maximum number of sessions allowed for users depends upon the software license installed on the device.*

Syntax

get

`get active-user`

Keywords and Variables

None.

address

Description: Use the **address** commands to define entries in the address book of a security zone.

An *address book* is a list containing all addresses, address groups, and domain names defined for a security zone. You use address book entries to identify addressable entities in policy definitions.

Syntax

get

```
get address zone [ group name_str | name name_str ]
```

set

```
set address zone name_str  
{  
  fqdn |  
  ip_addr/mask  
}  
[ string ]
```

unset

```
unset address zone name_str
```

Keywords and Variables

Variable Parameters

<i>zone</i>	The name of the security zone. The default security zones to which you can bind an address book include Trust, Untrust, Global, DMZ, V1-Trust, V1-Untrust, and V1-DMZ. You can also assign address book entries to user-defined zones. For more information on zones, see "Zone Names" on page B-1 .
<i>fqdn</i>	The fully-qualified domain name of the host.
<i>ip_addr/mask</i>	The address and subnet mask identifying an individual host or a subnet.
<i>string</i>	A character string containing a comment line.

Example: The following commands create address book entries named "Local_Net" and "Outside_Net".

```
set address trust Local_Net 10.1.1.0/24 "New_York_Subnet"  
set address untrust Outside_Net 1.1.12.1/24 "London_Subnet"
```

group

```
get address zone group name_str
```

group	The name of a group of address book entries. You can use an address group in a security policy definition to specify multiple addresses. (Create address groups using the set group address command.)
--------------	--

Example: The following command displays information for an address group named Sales_Group:

```
get address trust group HTTP_Servers
```

name

name <i>name_str</i>	The name of an individual address book entry. You can use an address group in a security policy definition to specify a single address.
-----------------------------	---

admin

Description: Use the **admin** commands to configure or display administrative parameters for the NetScreen device. These parameters determine the following:

- Characteristics for each admin user, such as password and privilege level
- How the device performs admin user authentication
- Ways that admin users can access the device
- Which IP address to use for administering the device from the web
- Which port the device uses to detect configuration changes made through the web
- Whether the device automatically emails generated alerts and traffic alarms
- Whether the device is enabled for reset

Syntax

clear

```
clear [ cluster ] admin { all | name name_str }
```

get

```
get admin  
[  
  auth [ banner | settings ] |  
  current-user |  
  manager-ip |  
  ssh all |  
  user [ cache | login ]  
]
```

set

```
set admin
{
  access attempts number |
  auth
  {
    banner { console | telnet } login string |
    server name_str |
    timeout number |
  } |
  device-reset |
  format { dos | unix } |
  hw-reset |
  mail
  {
    alert |
    mail-addr1 name_str | mail-addr2 name_str |
    server-name { ip_addr | name_str } |
    traffic-log
  } |
  manager-ip ip_addr [ mask ] |
  name name_str |
  password [ pswd_str | restrict length number ] |
  port port_num |
  privilege { get-external | read-write } |
  root access console |
  ssh
  {
    password { disable | enable } username name_str |
    port port_num
  } |
  telnet port port_num |
  user name_str password pswd_str [ privilege { all | read-only } ]
}
```

unset

```
unset admin
{
  access attempts |
  auth
  {
    banner { console | telnet } login |
    server |
    timeout
  } |
  device-reset |
  format |
  hw-reset |
  mail
  { alert | mail-addr1 | mail-addr2 | server-name | traffic-log } |
  manager-ip { ip_addr | all } |
  name |
  password [ restrict-length ] |
  port |
  privilege |
  root access |
  ssh [ port ] |
  telnet { access tunnel | port } |
  user name_str
}
```

Keywords and Variables

access attempts

```
set admin access attempts number  
unset admin access attempts
```

access attempts Specifies the number (1 - 255) of unsuccessful login attempts allowed before the device closes the Telnet connection. The default is 3.

Example: The following command sets the number of allowed unsuccessful login attempts to 5:

```
set admin access attempts 5
```

alert

```
set admin mail alert
```

alert Collects system alarms from the device for sending to an email address.

alert

```
clear admin all
```

all Clears all admin user profiles.

auth

```
get admin auth [ banner | settings ]
set admin auth banner { console | telnet } login string
set admin auth server name_str }
set admin auth timeout number
unset admin auth { banner { console | telnet } login | server | timeout }
```

auth

Configures admin authentication settings for the NetScreen device.

- **banner** Specifies the banner (*string*) displayed during login through the console port (**console**) or a Telnet session (**telnet**).
- **server** The name of the authentication server used for authenticating admin users.
- **timeout** Specifies the length of idle time (in minutes) before the NetScreen device automatically closes the web administrative session. The value can be up 999 minutes. A value of 0 specifies no timeout. (Telnet admin sessions time out after the console timeout interval expires. You set this interval using the **set console timeout** command.)

Example2: The following commands create two login banners:

- "Hyperterminal Management Console," displayed at the start of new console admin sessions
- "Telnet Login Here," displayed at the start of new Telnet admin sessions

```
set admin auth banner console login "Hyperterminal Management Console"
set admin auth banner telnet login "Telnet Login Here"
```

cache

```
clear [ cluster ] admin user cache
get admin user cache
```

cache

Clears or displays the memory cache containing all current remote administrative users.

cluster

```
clear cluster admin user { cache | login }
```

cluster Propagates the **clear** operation to all other devices in a NSRP cluster.

Example: The following command clears remote administrative users from the cache and propagates this change to other devices in a NSRP cluster:

```
clear cluster admin user cache
```

current-user

```
get admin current-user
```

current-user Displays the user for the current administrative session.

device-reset

```
set admin device-reset  
unset admin device-reset
```

device-reset Enables device reset for asset recovery.

format

```
set admin format { dos | unix }  
unset admin format
```

format Determines the format (**dos** or **unix**) used when the NetScreen device generates the configuration file. On some NetScreen platforms, you can download this file to a TFTP server or PCMCIA card using the CLI, or to a local directory using WebUI.

hw-reset

```
set admin hw-reset
unset admin hw-reset
```

hw-reset Enables hardware reset for asset recovery.

login

```
clear [ cluster ] admin user login
get admin user login
```

login Clears or displays all current administrative users.

mail

```
set admin mail { ... }
unset admin mail { ... }
```

mail Enables email for sending alerts and traffic logs.

Example: The following command configures the email address *john@abc.com* to receive updates concerning administrative issues:

```
set admin mail mail-addr1 john@abc.com
```

mail-addr1

```
set admin mail mail-addr1 name_str
```

mail-addr1 *name_str* Sets the first e-mail address (such as **chris@acme.com**) for sending alert and traffic logs.

mail-addr2

```
set admin mail mail-addr2 name_str
```

mail-addr2 *name_str* Sets the secondary email address for sending alert and traffic logs.

Example: The following command configures the secondary email address *pat@acme.com* to receive updates concerning administrative issues:

```
set admin mail mail-addr2 pat@acme.com
```

manager-ip

```
get admin manager-ip  
set admin manager-ip ip_addr [ mask ssh [ port ]  
unset admin manager-ip { ip_addr | all }
```

manager-ip Restricts management to a host or a subnet. The default **manager-ip** address is 0.0.0.0, which allows management from any workstation. All NetScreen devices allow you to specify up to six hosts or subnets at once.

Note: The **manager-ip** address must be unique, and different from the physical IP address of the management interface.

Example: The following command restricts management to a single host with IP address 10.1.10.100:

```
set admin manager-ip 10.1.10.100 255.255.255.255
```

name

```
set admin name name_str  
unset admin name
```

name The login name (*name_str*) of the root user for the NetScreen device. The maximum length of the name is 31 characters, including all symbols except ?. The name is case-sensitive.

password

```
set admin password pswd_str  
unset admin password
```

password Specifies the password (*pswd_str*) of the root user. The maximum length of the password is 31 characters, including all symbols except the special command character "?."

port

```
set admin port port_num  
unset admin port
```

port Sets the port number (*port_num*) for detecting configuration changes when using the web. Use any number between 1024 and 32767, or use the default port number (80). Changing the admin port number might require resetting the device (see the **reset** command).

privilege

```
set admin privilege ( get-external | read-write )
```

privilege Defines the administrative privilege level:

- **get-external** Instructs the NetScreen device to obtain the admin user privileges externally from the RADIUS server.
- **read-write** Gives the RADIUS administrator read-write privileges, and ignores the privilege returned from the RADIUS server.

restrict length

```
set admin password restrict length number  
unset admin password restrict length
```

restrict length Sets the minimum password length of the root admin. The password length can be any number from 1 to 31.

root access console

```
set admin root access console
unset admin root access console
```

root access console Restricts the root admin to logging in to the device through the console only.

server-name

```
set admin mail server-name ip_addr
```

server-name The IP address or name of the Simple Mail Transfer Protocol (SMTP) server. This server receives email notification of system alarms and traffic logs.

Example: The following command specifies a SMTP server at IP address 10.1.10.10:

```
set admin mail server-name 10.1.10.10
```

settings

```
get admin auth settings
```

settings Displays admin authentication settings, including the current timeout setting and the admin user type (local or remote).

ssh

```
get admin ssh all
set admin ssh password { disable | enable } username name_str
set admin ssh password port port_num
unset admin ssh [ port ]
```

- ssh** Provides access to the Secure Shell (SSH) utility. SSH allows you to administer NetScreen devices from an Ethernet connection or a dial-in modem, thus providing secure CLI access over unsecured channels.
- **all** Displays the SSH PKA (Public Key Authentication) information for each admin.
 - **password** Sets the password for the user that establishes the SSH session. The **enable | disable** switch enables or disables password authentication. **username** *name_str* specifies the admin user name.
 - **port** *port_num* Specifies the logical SSH port through which the communication occurs. The default is port 22. Unsetting the port resets the SSH port to the default.

telnet

```
set admin telnet port port_num
unset admin telnet port
```

- telnet port** Provides CLI access through a Telnet connection. The acceptable range of *port_num* is 1024 - 32767.

traffic-log

```
set admin mail traffic-log
unset admin mail traffic-log
```

- traffic-log** Generates a log of network traffic handled by the NetScreen device. The traffic log can contain a maximum of 4,096 entries. The NetScreen device sends a copy of the log file to each specified e-mail address (see [mail-addr1](#) and [mail-addr2](#)). This happens when the log is full, or every 24 hours, depending upon which occurs first.

user

```
get admin user [ cache | login ]
set admin user name_str password pswd_str [ privilege { all | read-only } ]
unset admin user name_str
```

user Creates or displays a non-root administrator (super-administrator or sub-administrator). The maximum user name length is 31 characters, including all symbols except ?. The user name is case-sensitive. The **privilege** switch determines the privilege level of the user (**all** or **read-only**).

Example: The following command creates a non-root administrator named "rsmith", with password "swordfish":

```
set admin user rsmith password swordfish privilege all
```

Defaults

The default admin name and password are netscreen.

The default number of access attempts is 3.

The default manager-ip is 0.0.0.0, and the default subnet mask is 255.255.255.255.

The default privilege for a super-administrator is read-only.

The default admin port is 80.

The default mail alert setting is off.

The default for device reset is on.

alarm

Description: Use the **alarm** commands to set or display alarm parameters.

The alarm parameters determine when the device generates alarm messages, and the amount and type of information contained in the messages.

Syntax

clear

```
clear [ cluster ] alarm traffic
      [ policy pol_num1 [ -pol_num2 ] ]
      [ end-time string ]
```

get

```
get alarm
{
  threshold |
  traffic
    [ policy { pol_num1 [ -pol_num2 ] } ]
    [ service name_str ]
    [ src-address ip_addr ] [ dst-address ip_addr ]
    [ detail
      [ start-time string ] [ end-time string ]
      [ minute | second
        [ threshold number [ -number ] ]
        [ rate number [ -number ] ]
      ]
    ] |
}
```

set

```
set alarm threshold
{
  cpu number |
  memory number |
  session { count number | percent number }
}
```

unset

```
unset alarm threshold { CPU | memory | session }
```

Keywords and Variables

cluster

```
clear cluster alarm traffic [ ... ]
```

cluster Propagates the **clear** operation to all other devices in a NSRP cluster.

Example: The following command clears the alarm table entries for policy 4 and propagates the change to other device in a NSRP cluster:

```
clear cluster alarm traffic policy 4
```

detail

```
get alarm traffic [ ... ] detail [ ... ]
```

detail Displays detailed information for each policy, including all traffic alarm entries that occurred under the policy. If you omit this option, the output contains only general information and the time of the most recent alarm for each policy.

Example: The following command displays event alarm entries or traffic alarm entries that occur on or after January 1, 2003:

```
get alarm traffic detail start-time 01/01/2003
```

end-time | start-time

```
clear [ cluster ] alarm traffic policy [ ... ] end-time number
```

```
get alarm traffic [ ... ] end-time number
```

```
get alarm traffic [ ... ] start-time number
```

start-time The **start-time** option displays event alarm entries or traffic alarm entries that occurred at or before the time specified. The **end-time** option displays event alarm entries or traffic alarm entries that occurred at or after the time specified. The format for *string* is *mm/dd[/yy-hh:mm:ss]*
end-time You can omit the year (the current year is the default), or express the year using the last two digits or all four digits. The hour, minute, and second are optional. The delimiter between the date and the time can be a dash or an underscore:
 12/31/2002-23:59:00
 12/31/2002_23:59:00

Example: The following command performs a detailed display of traffic alarm entries at (or after) 11:59pm, December 31, 2003 and at or before 12:00am, December 31, 2004:

```
get alarm traffic detail start-time 12/31/2003-23:59:00 end-time  
                  12/31/2004-24:00:00
```

policy

```
clear [ cluster ] alarm traffic policy pol_num1 [ -pol_num2 ] [ ... ]
```

```
get alarm traffic policy pol_num
```

policy Displays traffic alarm entries for a policy specified by its ID number or for several policies specified by a range of ID numbers. The ID number can be any value between 0 and the total number of established policies. To define a range, enter the starting and ending ID numbers as follows: *pol_num1-pol_num2*

Example: The following command clears the entries for policy 2 in the alarm table:

```
clear alarm traffic policy 2
```

second | minute

```
get alarm traffic [ ... ] detail
```

- second | minute** Displays traffic alarm entries for policies with threshold settings at bytes per second or bytes per minute.
- The **rate** *number* [-*number*] option displays traffic alarm entries for policies with a flow rate at a specified value or within a specified range.
 - The **threshold** *number* [-*number*] option displays traffic alarm entries for policies with a threshold at a specified value or within a specified range.

Example: The following command displays traffic alarm entries for policies with threshold settings at bytes per second:

```
get alarm traffic detail second
```

service

```
get alarm traffic [ ... ] service name_str [ ... ]
```

- service** Displays traffic alarm entries for a specified service (*name_str*), such as TCP, ICMP, or FTP. (To display all services, make the *name_str* value **Any**.) The name does not have to be complete; for example, both **TC** and **CP** are recognized as **TCP**. Although you cannot specify a Service group, note that because **TP** is recognized as **FTP**, **HTTP**, and **TFTP**, entering **TP** displays traffic alarm entries for all three of these Services.

Example: The following command displays traffic alarm entries for the HTTP service:

```
get alarm traffic service http
```

src-address | dst-addr

```
get alarm traffic [ ... ] src-address ip_addr [ ... ]
get alarm traffic [ ... ] dst-address ip_addr [ ... ]
```

src-address Displays traffic alarm entries originating from a specified IP address (*ip_addr*) or from a specified direction, such as **inside_any** or **outside_any**.

dst-address Displays traffic alarm entries destined for a specified IP address (*ip_addr*) or for a specified direction, such as **inside_any** or **outside_any**.

Example: The following command displays traffic alarm entries originating from IP address 10.1.9.9 and destined for IP address 1.1.10.10:

```
get alarm traffic src-address 10.1.9.9 dst-address 1.1.10.10
```

threshold

```
get alarm threshold
get alarm traffic [ ... ] threshold number [ -number ]
set alarm threshold { ... }
unset alarm threshold { CPU | memory | session }
```

threshold Displays traffic alarm entries for policies with threshold settings at a specified value or within a specified range.

- **cpu** *number* sets the cpu threshold.
- **memory** *number* sets the memory threshold.
- **session** sets the session threshold. The **count** *number* option specifies how many sessions can exist before the device generates an alarm. The **percent** *number* option specifies what percentage of the session limit is allowable before the device generates an alarm.

Example: The following command sets the session limit threshold to 75,000 sessions:

```
set alarm threshold session count 75000
```

traffic

```
clear [ cluster ] alarm traffic [ ... ]  
get alarm traffic [ ... ]
```

traffic Specifies traffic alarm entries.

Example: The following command performs a detailed display of traffic alarm entries originating from IP address 10.1.9.9 and destined for IP address 1.1.10.10:

```
get alarm traffic src-address 10.1.9.9 dst-address 1.1.10.10 detail
```

alias

Description: Use the **alias** commands to create, remove, or list aliases. An *alias* is a named variable containing the initial characters of a CLI command. After creating an alias, you can use it to execute the represented command.

Syntax

get

```
get alias
```

set

```
set alias name_str string
```

unset

```
unset alias name_str
```

Keywords and Variables

Variable Parameters

<i>name_str</i>	The name of the CLI command alias.
<i>string</i>	The CLI command to which you assign the alias.

Example: The following commands create an alias representing the **get interface ethernet1/1** command, then execute the command using the alias:

```
set alias int_1 "get interface ethernet1/1"  
int_1
```

Description: Use the **all** command to return all configuration settings to the factory default values.

Syntax

```
unset all
```

Keywords and Variables

None.

Example

In the following example, you reset the device to its factory default settings and reset the device.

1. Execute the **unset all** command.

```
unset all
```

The following prompt appears: "Erase all system config, are you sure y / [n]?"

2. Press the **Y** key. This action returns the system configuration to the factory default settings.
3. Execute the **reset** command.

```
reset
```

The following prompt appears: "Configuration modified, save? [y] / n"

4. Press the **N** key. This action generates the following prompt: "System reset, are you sure? y / [n] n"
5. Press the **Y** key. This action restarts the system. The device now has its original factory default settings.

arp

Description: Use the **arp** commands to create, remove, or list interface entries in the Address Resolution Protocol (ARP) table of the NetScreen device.

Syntax

clear

```
clear [ cluster ] arp [ ip_addr | all ] |
```

get

```
get arp [ all | asic id_num ]
```

set

```
set arp
{
  ip_addr mac_addr interface |
  age number |
  always-on-dest
}
```

unset

```
unset arp
{
  ip_addr [ interface ] |
  age |
  always-on-dest
}
```

Keywords and Variables

Variable Parameters

set arp *ip_addr mac_addr interface*

ip_addr The IP address for the interface in the ARP table entry.

mac_addr The MAC address for the interface in the ARP table entry.

interface The name of the ARP interface in the ARP table entry. For more information on interfaces, see ["Interface Names" on page A-I](#).

all

get arp *all*

age Lists all current ARP entries for every existing vsys.

asic

get asic *id_num*

asic Lists all current ARP entries for every a specified ASIC chip (*id_num*).

age

set arp age *number*

age Sets the age-out value (in seconds) for ARP entries.

always-on-dest

set arp always-on-dest

always-on-dest Directs the NetScreen device to send an ARP request for any incoming packet with a heading containing a MAC address not yet listed in the device's MAC address table. This may be necessary when packets originate from server load-balancing (SLB) switches or from devices using the Hot Standby Router Protocol/Virtual Router Redundancy Protocol (HSRP/VRRP).

cluster

clear [cluster] arp

cluster Propagates the **clear** operation to all other devices in a NSRP cluster.

attack

Description: Use the **attack** commands to view and define attack objects, attack object groups, and the attack object database server settings.

Note: This command is available only if advanced mode is installed on the device.

Syntax

get

```
get attack
[
  name_str |
  anomaly [ sort-by { id | name } ] |
  db |
  group [ sort-by { def-type | name } ] |
  id id_num |
  [ signature ] sort-by { def-type | id | name | type }
]
```

set

```
set attack
{
  name_str
  {
    ftp-command string |
    ftp-username string |
    http-url-parsed string |
    smtp-from string |
    smtp-header-from string |
    smtp-header-to string |
    smtp-rcpt string
  }
  severity { info | low | medium | high | critical | number } |
}
```

```
db
{
  ca-idx id_num |
  cert-subject string |
  mode { notification | update } |
  schedule
  {
    daily hh:mm |
    monthly number hh:mm |
    weekly day hh:mm
  } |
  server url_str
} |
group name_str1 [ add name_str2 ] |
}
```

unset

```
unset attack
{
  name_str |
  db { mode | schedule | server url_str } |
  group name_str1 [ remove name_str2 ]
}
```

Keywords and Variables

Variable Parameter

```
get attack name_str
set attack name_str ftp-command string severity string
set attack name_str ftp-username string severity string
set attack name_str http-url-parsed string severity string
set attack name_str smtp-from string severity string
set attack name_str smtp-header-from string severity string
set attack name_str smtp-header-to string severity string
set attack name_str smtp-rcpt string severity string
unset attack name_str
```

name_str

Defines the attack object name.

Specifies one of the following contexts for Deep Inspection search and defines the signature string for which the Deep Inspection module searches:

- **ftp-command** *string*
- **ftp-username** *string*
- **http-url-parsed** *string*
- **smtp-from** *string*
- **smtp-header-from** *string*
- **smtp-header-to** *string*
- **smtp-rcpt** *string*

- **severity** Defines the severity level of the attack. You can specify any of the following levels: **info**, **low**, **medium**, **high**, **critical**.

Example: The following command creates an FTP-command attack object named "rootuser", defines its signature, and specifies its severity level as high:

```
set attack rootuser ftp-username root severity high
```

anomaly

```
get attack anomaly [ sort-by { id | name } ]
```

anomaly Specifies the attack object as an anomaly.

sort-by Indicates the organization for the display of protocol anomalies in the local database—either numerically by **id** or alphabetically by **name**.

db

```
get attack db
set attack db ca-idx id_num
set attack db cert-subject string
set attack db mode { notification | update }
set attack db schedule { daily hh:mm | monthly number hh:mm | weekly day hh:mm }
set attack db server url_str
unset attack db { ca-idx | cert-subject | mode | schedule | server url_str }
```

db Specifies the attack object database server. On NetScreen devices that support virtual systems, you must set this command at the root level.

- **mode** Selects either **notification** or **update** as the mode for checking and updating the attack object database. The **notification** method automatically checks the attack object database server at user-defined times and notifies the NetScreen admin if the database on the server is more recent than the one on the NetScreen device. (If the data on the server is more recent, a notice appears on the Home page in the WebUI, and in the CLI after you log in to the NetScreen device.) The **update** method automatically checks the attack object database server at user-defined times and automatically updates the database on the NetScreen device if it determines that the database on the server is more recent than the one on the NetScreen device.
- **schedule** *string* Sets the time for automatically checking the attack object database server and updating the attack object database on the NetScreen device. You can set a daily, monthly or weekly schedule.
- **server** *url_str* Defines the URL of the attack object database server. Unsetting the attack object database server stops the NetScreen device from automatically checking the server or from updating the local database.

Example: The following commands define the URL of the attack object database server and set a schedule to check the server automatically and then notify the NetScreen admin when the database on the server is more recent than that on the NetScreen device:

```
set attack db server http://help.netscreen.com/attacks
set attack db schedule daily 07:00
set attack db mode notification
```

Example: The following commands specify CA certificate ID 0001 and define the text string that must appear in the subject name field of the certificate sent by the attack object database server:

```
set attack db ca-idx 0001
set attack db cert-subject CN=SecurityCA,OU=Security,O=Juniper,L=Sunnyvale,
    ST=CA,C=US
```

group

```
get attack group [ sort-by { def-type | name } ]
set attack group name_str1 [ add name_str2 ]
unset attack group name_str1 [ remove name_str2 ]
```

group Specifies an attack object group.

sort-by Indicates the organization for the display of attack groups from the local database:

- **def-type:** Organizes the attack group display by the definition type of the attack group, displaying first the predefined attack groups in alphabetical order and then the user-defined groups.
- **name:** Organizes the attack group display alphabetically by attack group names.

name_str specifies a name for the creation, deletion, or modification of an attack group. The keywords **add** and **remove** indicate that an attack is being added to or is being removed from the specified group.

Example: The following command displays all the attack groups on the NetScreen device by name in alphabetical order:

```
get attack group sort-by name
```

id

```
get attack id id_num
```

id Specifies the ID number of an attack object in the local database.

Example: The following command displays the attack object with ID number 500 in the NetScreen device:

```
get attack id 500
```

signature

```
get attack signature [ sort-by { def-type | id | name } ]
```

signature Displays stateful signature attack objects currently stored in the local database.

- **sort-by:** Specifies the organizational display of attack objects by one of the following attributes:

- **def-type:** Organizes the stateful signature attack object display by the definition type of the attack object, displaying first the predefined attack objects in alphabetical order and then the user-defined objects.
- **id:** Organizes the stateful signature attack object display numerically by ID number.
- **name:** Organizes the stateful signature attack object display alphabetically by attack name.

Example: The following command displays signature attack objects on the NetScreen device alphabetically by name:

```
get attack signature sort-by name
```

sort-by

```
get attack sort-by { def-type | id | name | type }
```

- sort-by** Specifies the organizational display of attack objects in the local database by one of the following attributes:
- **def-type:** Organizes the attack object display by the definition type of the attack object, displaying first the predefined attack objects in alphabetical order and then the user-defined objects.
 - **id:** Organizes the attack object display numerically by ID number.
 - **name:** Organizes the attack object display alphabetically by attack name.
 - **type:** Organizes the attack object display alphabetically by protocol type, and then within each protocol type by attack name.

Example: The following command displays all attack objects in the NetScreen device organized numerically:

```
get attack sort-by id
```


attack-db

Description: Use the **attack-db** commands to perform attack object database updates and checking.

Note: This command is available only if advanced mode and the Deep Inspection db key is installed on the device.

Syntax

exec

exec attack-db { check | update }

Keywords and Variables

check

exec attack-db check

check	Immediately checks if the attack object database on the server is more recent than the one on the NetScreen device.
--------------	---

update

exec attack-db update

update	Updates the attack object database on the NetScreen device immediately with the database stored on the attack object database server.
---------------	---

audible-alarm

Description: Use the **audible-alarm** command to activate the audible alarm feature.

Syntax

get

```
get audible-alarm
```

set

```
set audible-alarm  
    { all | battery | fan-failed | power-failed | temperature }
```

unset

```
unset audible-alarm  
    { all | battery | fan-failed | power-failed | temperature }
```

Arguments

- audible-alarm** Enables or disables the audible alarm to announce hardware failure events.
- **all** Enables or disables the audible alarm in the event of a fan failure, a interface module failure, a power supply failure, or a temperature increase above an admin-defined threshold.
 - **battery** Enables or disables the audible alarm in the event of battery failure.
 - **fan-failed** Enables or disables the audible alarm in the event of a fan failure.
 - **module-failed** Enables or disables the audible alarm in the event of an interface module failure.
 - **power-failed** Enables or disables the audible alarm in the event of a power supply failure.
 - **temperature** Enables or disables the audible alarm if the temperature rises above an admin-defined threshold.

Example: To enable the audible alarm to sound in the event that one or more of the fans in the fan assembly fails:

```
set audible-alarm fan-failed
```

Description: Use the **auth** commands to specify a user authentication method.

The four available methods include:

- A built-in database
- A RADIUS server
- SecurID
- Lightweight Directory Access Protocol (LDAP)

Note: If the NetScreen device uses SecurID to authenticate users, and communication problems occur with the ACE server, clear the current SecurID shared secret from the device (and the server) by executing the **clear node_secret** command.

Syntax

clear

```
clear [ cluster ] auth
[
  history |
  queue |
  table [ id id_num | ip ip_addr ]
]
```

get

```
get auth
[
  banner |
  history [ id id_num | ip ip_addr ] |
  queue |
  settings |
  table [ id id_num | ip ip_addr ]
]
```

set

```
set auth
{
  banner { ftp | http | telnet }
    { fail string | login string | success string }
  default auth server name_str
}
```

unset

```
unset auth
{
  banner { ftp | http | telnet }
    { fail | login | success }
  default auth server
}
```

Arguments

banner

```
get auth banner
set auth banner { ftp | http | telnet }
unset auth banner { ftp | http | telnet }
```

- banner** Defines or displays firewall banners. The NetScreen device uses these banners to report success or failure of login requests.
- **ftp** Reports on the success or failure of FTP login requests.
 - **http** Reports on the success or failure of HTTP login requests.
 - **telnet** Reports on the success or failure of Telnet login requests.
 - **fail** *string* Specifies a message string to display a login attempt fails.
 - **login** *string* Specifies a message string to display when a login attempt occurs.
 - **success** *string* Specifies a message string to display when a login attempt is successful.

Example: The following command defines a banner for a failed FTP login attempt:

```
set auth banner ftp fail "FTP login attempt failed"
```

cluster

```
clear [ cluster ] auth [ ... ]
```

- cluster** Propagates the **clear** operation to all other devices in a NSRP cluster.

default

```
set auth default auth server name_str
unset auth default auth server
```

default auth server Specifies a default firewall authentication server (*name_str*). The NetScreen device uses this server when a security policy does not explicitly identify a particular authentication server.

Example: The following command identifies the default authentication server (Auth_Server):

```
set auth default auth server Auth_Server
```

history

```
clear [ cluster ] auth history
get auth history [ id id_num | ip ip_addr ]
```

history Clears or displays the history of users authenticated through the NetScreen device.

queue

```
clear [ cluster ] auth queue
get auth queue
```

queue Clears or displays the internal user authentication queue.

settings

```
get auth settings
```

settings Displays default user authentication server settings. (This option yields the same display as the **get auth** command.)

table

```
clear [ cluster ] auth table [ id id_num | ip ip_addr ]  
get auth table [ id id_num | ip ip_addr ]
```

table Clears entries from the user authentication table (thus forcing reauthentication), or displays such entries. Entries in the user authentication table can represent:

- Users currently authenticated
- Users currently undergoing authentication
- Users denied authentication

Without parameters (described below), the **table** option clears or displays all table entries.

- **id** *id_num* Clears or displays a particular entry by ID (*id_num*).
- **ip** *ip_addr* Clears or displays all entries with a common source IP address (*ip_addr*).

Examples: The following command clears entry 7 from the user authentication table:

```
clear auth table id 7
```

The following command displays authentication details from a table entry with source IP 10.1.10.10:

```
get auth table ip 10.1.10.10
```


auth-server

Description: Use the **auth-server** commands to configure the NetScreen device for user authentication with a specified authentication server. Admins, policies, VPN tunnel specifications, and XAuth configurations use these server specifications to gain access to the appropriate resources.

Syntax

get

```
get auth-server
{
  string |
  all |
  id id_num
}
```

set

```
set auth-server name_str
{
  account-type { [ admin ] | [ auth ] [ l2tp ] [ xauth ] } |
  backup1 { ip_addr | name_str } |
  backup2 { ip_addr | name_str } |
  id id_num |
  ldap
  {
    cn name_str |
    dn name_str |
    port port_num |
    server-name { ip_addr | name_str }
  } |
  radius
  {
```

```
    port port_num |
    secret shar_secret |
    timeout number
  } |
  securid
  {
    auth-port port_num |
    duress number |
    encr id_num |
    retries number |
    timeout number
  } |
  server-name { ip_addr | name_str } |
  timeout number |
  type { ldap | radius | securid }
}
```

unset

```
unset auth-server
{
  name_string
  [
    account-type
    {
      admin |
      [ auth ] [ ike ] [ l2tp ] [ xauth ]
    }
    backup1 | backup2 |
    radius { port | timeout } |
    timeout |
    type
  ]
  id id_num
}
```

Keywords and Variables

Variable Parameter

```
set auth-server name_str [ ... ]
```

name_str Identifies the object name of the authentication server.

Example: The following command creates a server object name (radius1) and specifies type RADIUS:

```
set auth-server radius1 type radius
```

account-type

```
set auth-server name_str account-type { [ admin ] | [ auth ] [ l2tp ] [ xauth ] }
```

account-type Specifies the types of users authenticated by the server (*name_str*).

- **admin** specifies admin users.
- **auth** specifies authentication users.
- **l2tp** specifies Layer 2 Tunneling Protocol (L2TP) users.
- **xauth** specifies XAuth users.

You can define a user as a single user type—an admin user, an authentication user, an L2TP user, or an XAuth user. You can combine auth, L2TP, and XAuth user types to create an auth-L2TP user, an auth-XAuth user, an L2TP-XAuth user, or an auth-L2TP-XAuth user. You cannot combine an admin user with another user type.

all

```
get auth-server all
```

all Specifies all configured authentication servers.

backup1 | backup2

```
set auth-server name_str { backup1 { ip_addr | name_str } | backup2 { ip_addr |  
    name_str } }  
unset auth-server name_str { backup1 | backup2 }
```

- | | |
|----------------|---|
| backup1 | The IP address or DNS name of the primary backup authentication server for an LDAP, RADIUS, or SecurID server type. |
| backup2 | The IP address or DNS name of the secondary backup authentication server for an LDAP or RADIUS server type. SecurID does not support more than one backup server. |

Example: With the following commands, you first create a RADIUS authentication server object named “radius1” at IP address 10.1.1.50. It stores authentication user accounts. Then you define a primary backup server at 10.1.1.51 and a secondary backup server at 10.1.1.52:

```
set auth-server radius1 server-name 10.1.1.50  
set auth-server radius1 type radius  
set auth-server radius1 account-type auth  
set auth-server radius1 backup1 10.1.1.51  
set auth-server radius1 backup2 10.1.1.52
```

id

```
get auth-server id id_num  
set auth-server name_str id id_num  
unset auth-server id id_num
```

- | | |
|-----------|---|
| id | The user-defined identification number (<i>id_num</i>) of the authentication server. If you do not define an ID number, the NetScreen device creates one automatically. |
|-----------|---|

Example: The following command creates an identification number (200) for the authentication server radius1:

```
set auth-server radius1 id 200
```

ldap

```
set auth-server name_str ldap { ... }
```

ldap

Configures the NetScreen device to use an LDAP server for authentication.

- **cn** *name_str* The Common Name identifier used by the LDAP server to identify the individual entered in a LDAP server. For example, an entry of "uid" means "user ID" and "cn" means "common name".
- **dn** *name_str* The Distinguished Name identifier is the path used by the LDAP server before using the common name identifier to search for a specific entry (for example, c=us;o=juniper, where "c" stands for "country", and "o" for "organization").
- **port** *port_num* Specifies the port number to use for communication with the LDAP server. The default port number for LDAP is 389.
- **server-name** *name_str* The IP address or DNS name of the LDAP server.

Example: For an example of this option, see ["Defining an LDAP Server Object" on page 51](#).

radius

```
set auth-server name_str radius { ... }  
unset auth-server name_str radius { port | timeout }
```

radius

Configures the NetScreen device to use a RADIUS server for authentication.

- **port** *port_num* The port number on a RADIUS server to which the NetScreen device sends authentication requests. The default port number is 1645. You can change the default port number to any number between 1024 and 65535.
- **secret** *shar_secret* Specifies the RADIUS shared secret (*shar_secret*) that is shared between the NetScreen device and the RADIUS server. The NetScreen device uses this secret to encrypt the user's password that it sends to the RADIUS server.
- **timeout** *number* The interval (in seconds) that the NetScreen device waits before sending another authentication request to the RADIUS server if the previous request does not elicit a response. The default is three seconds.

Example: For an example of these options, see ["Defining a RADIUS Server Object" on page 50](#).

securid

```
set auth-server name_str securid
{
  auth-port port_num |
  duress number |
  encr id_num |
  retries number |
  timeout number
}
```

securid

Configures the NetScreen device to use a SecurID server for authentication.

- **auth-port** *port_num* Specifies the port number to use for communications with the SecurID server. The default SecurID port number is 5500.
- **duress** { **0** | **1** } If the SecurID server is licensed to use duress mode, a value of 0 deactivates it and 1 activates it. When duress mode is activated, a user can enter a special duress PIN number when logging in. The NetScreen device allows the login, but sends a signal to the SecurID server, indicating that someone is forcing the user to login against his or her will. The SecurID auth server blocks further login attempts by that user until he or she contacts the SecurID server admin.
- **encr** { **0** | **1** } Specifies the encryption algorithm for SecurID network traffic. A value of 0 specifies SDI, and 1 specifies DES. Juniper Networks recommends the default encryption type DES.
- **retries** *number* Specifies the number of retries between requests for authentication.
- **timeout** *number* Specifies the length of time (in seconds) that the NetScreen device waits between authentication retry attempts.

Example: For an example of this option, see [“Defining a SecurID Server Object” on page 50](#).

server-name

```
set auth-server name_str server-name { ip_addr | name_str }
```

server-name

The IP address or DNS name of the authentication server.

timeout

```
set auth-server name_str timeout number  
unset auth-server name_str timeout
```

timeout Specifies how many minutes must elapse after termination of an authentication, L2TP, or XAuth user's last session before the user needs to reauthenticate. The default timeout value is 10 minutes, and the maximum setting is 255 minutes. If the user initiates a new session before the countdown reaches the timeout threshold, he does not have to reauthenticate himself and the timeout countdown resets.

If the user is an admin user, this setting specifies how many minutes of inactivity must elapse before the NetScreen device times out and closes an admin session. The default is 10 minutes and the maximum is 1000 minutes.

Example: For an example of this option, see ["Defining a SecurID Server Object" on page 50](#).

type

```
set auth-server name_str type { ldap | radius | securid }
```

type Specifies the type of authentication server—LDAP, SecurID or RADIUS. The **unset** command sets **type** to **radius**.

Example: For an example of this option, see ["Defining a RADIUS Server Object" on page 50](#).

Defining a RADIUS Server Object

The following commands define an auth-server object for a RADIUS server:

```
set auth-server radius1 type radius
set auth-server radius1 account-type auth l2tp xauth
set auth-server radius1 server-name 10.1.1.50
set auth-server radius1 backup1 10.1.1.51
set auth-server radius1 backup2 10.1.1.52
set auth-server radius1 radius port 4500
set auth-server radius1 radius timeout 4
set auth-server radius1 radius secret A56htYY97kl
save
```

If you are using vendor-specific attributes, you must load the netscreen.dct file on the RADIUS server.

Defining a SecurID Server Object

The following commands define an auth-server object for a RADIUS server:

```
set auth-server securid1 type securid
set auth-server securid1 server-name 10.1.1.100
set auth-server securid1 backup1 10.1.1.110
set auth-server securid1 timeout 60
set auth-server securid1 account-type admin
set auth-server securid1 securid retries 3
set auth-server securid1 securid timeout 10
set auth-server securid1 securid auth-port 15000
set auth-server securid1 securid encr 1
set auth-server securid1 securid duress 0
save
```


Defining an LDAP Server Object

The following commands define an auth-server object for an LDAP server:

```
set auth-server ldap1 type ldap
set auth-server ldap1 account-type auth
set auth-server ldap1 server-name 10.1.1.150
set auth-server ldap1 backup1 10.1.1.151
set auth-server ldap1 backup2 10.1.1.152
set auth-server ldap1 timeout 40
set auth-server ldap1 ldap port 15000
set auth-server ldap1 ldap cn cn
set auth-server ldap1 ldap dn c=us;o=juniper;ou=marketing
save
```

The following command lists all auth-server settings:

```
get auth-server all
```

Description: Use the **av** commands to create an antivirus (AV) server, to identify an internal or external AV scanner, or to configure and view scanner operating parameters.

Juniper Networks NetScreen provides two AV scanning solutions:

- Internal AV scanning
- External AV scanning

With internal AV scanning, the AV scanner resides inside the NetScreen device as part of ScreenOS. Using a NetScreen device that supports internal AV simplifies deployment and management. It is a cost-effective choice for remote sites, small offices, retail outlets, and telecommuters. To configure the device for internal AV scanning, use the **scan-mgr** option (described below).

With external AV scanning, the AV scanner is a separate device to which the NetScreen device forwards traffic that requires scanning. Using a NetScreen device that supports one or more external AV scanners provides a flexible and scalable approach. You can begin with one AV scanner, but if the protected network grows, you can add more scanners (up to three total) to process increased traffic loads.

Syntax

clear

```
clear av { scan-mgr scanned bytes | statistics }
```

exec

```
exec av scan-mgr
{
  pattern-download tftp-server ip_addr file filename version number |
  pattern-update
}
```

get

```
get av { name_str | all | http | scan-mgr | statistics | trickling }
```

set

```
set av
{
  name_str
  {
    content { http [ timeout number ] | smtp [ timeout number ] } |
    server-name { ip_addr / dom_name } [ port port_num ]
  } |
  all
  {
    fail-mode { scanner threshold number | traffic permit } |
    max-connections number |
    resources number
  } |
  http { keep-alive | trickling { number1 number2 number3 | default } }
}
```

set (scan-mgr)

```
set av scan-mgr
{
  content { http | pop3 | smtp } [ timeout number ] |
  decompress-layer number |
  max-content-size { drop | number } |
  max-msgs { drop | number } |
  pattern-update-url url_str interval number
}
```

set (webmail)

```
set av http webmail
{
  enable |
  url-pattern-name string { args string | host string | path string }
}
```

unset

```
unset av
{
  string |
  all { fail-mode { scanner | traffic } | max-connections } |
  http |
  {
    keep-alive |
    trickling |
    webmail
    {
      enable |
      url-pattern-name string { args | host | path }
    }
  }
  scan-mgr
  {
    content { http | pop3 | smtp } [ timeout ] |
    decompress-layer |
    max-content-size [ drop ] |
    max-msgs [ drop ] |
    pattern-update-url
  }
}
```

Keywords and Variables

Variable Parameter

```
get av name_str
set av name_str [ ... ]
unset av name_str [ ... ]
```

name_str Identifies the object name of an AV scanner.

all

```
get av all
set av all { ... }
unset av all { ... }
```

all Enables and disables all defined AV scanners.

- **fail-mode** Sets the fail-mode parameters, which determine what the NetScreen device does when it loses connectivity with an AV scanner.
 - **scanner threshold** *number* Specifies the maximum number of times the device can attempt to connect with the AV scanner. After passing this threshold, the device changes state to fail-mode. The default value is 150.
 - **traffic permit** When enabled, directs the device to pass traffic after it enters fail-mode. The default setting for this parameter is disabled.
- **Note:** When **traffic permit** is disabled, the device blocks HTTP and SMTP traffic, even if a policy exists that permits AV checking.
- **max-connections** *number* Defines the maximum number of simultaneous TCP connections between the NetScreen device and the AV scanner.
- **resources** *number* The maximum percent of AV resources allowed per AV client. Possible settings range from 1 to 100, inclusive.

Example: The following command sets the maximum number of TCP connections allowed concurrently between the NetScreen device and the AV scanner.

```
set av all max-connections 10
```

Example: The following command instructs the NetScreen device to drop traffic if it cannot connect to the AV server:

```
unset av all fail-mode traffic
```

av statistics

```
clear av statistics
```

av statistics Clears the statistics concerning the total sessions per hour that the NetScreen device sent to the AV scanner, including the peak number sessions.

content

```
set av name_str content http [ timeout number ]
```

```
set av name_str content smtp [ timeout number ]
```

```
unset av name_str content http [ timeout ]
```

```
unset av name_str content smtp [ timeout ]
```

content Identifies the object name of the AV scanner and determines the kind of traffic to scan for viruses.

- **http** Enables AV scanning of HTTP traffic.
- **smtp** Enables AV scanning of SMTP traffic.

- **timeout number** Changes the timeout value for a Content Scanning Protocol (CSP) connection on a per-AV object basis. By default, a CSP connection times out after 180 seconds of inactivity. The range is 1 to 1800 seconds.

Example: The following command creates an AV object, classifies it as an HTTP traffic scanner, and defines the timeout as 120 seconds:

```
set av scanner1 content http timeout 120
```

http

```
get av http
set av http keep-alive
set av http trickling number1 number2 number3
set av http trickling default
set av http webmail { enable | url-pattern-name url_str { ... } }
unset av http keep-alive { ... }
```

http

Displays or sets HTTP configuration options for AV scanning.

- **keep-alive** Directs the NetScreen device to use the HTTP “keep-alive” connection option. Using this option prevents the device from sending a TCP FIN message to indicate termination of data transmission. (By default, the device uses the HTTP “close” connection option, which automatically sends the TCP FIN message.)
- **trickling** Configures the device for HTTP trickling, which automatically forwards specified amounts of unscanned HTTP traffic to the requesting HTTP client. Trickling prevents the client from timing out while the AV scanner is busy examining downloaded HTTP files.
 - *number1*: The minimum HTTP file size (in megabytes) needed to trigger the trickling action. The default value is 3 megabytes.
 - *number2*: The size (in megabytes) of each block of traffic sent to the AV scanner for scanning. The default value is 1 megabyte.
 - *number3*: The length (in bytes) of each trickle of unscanned HTTP traffic that the NetScreen device forwards to the client. The default value is 500 bytes.
 - **default** Resets the HTTP trickling parameters (*number1*, *number2*, and *number3*) to their default values.

- **webmail** Configures the NetScreen device for WebMail scanning.
 - **enable** Enables WebMail scanning only. **Note:** The device performs a full HTTP scan only if a policy exists that enables HTTP, and the **webmail** parameter is disabled.
 - **url-pattern-name** *url_str* Specifies a URL pattern identifying a certain type of WebMail to examine for virus patterns. When the URL matches all of the following parameters, the AV scanner performs a virus scan. The **args** *string* parameter specifies URL arguments, which begin with "?". The **host** *name_str* parameter specifies the host name included in the URL. The **path** *string* parameter specifies the download URL path for the type of webmail.

Example: The following command configures HTTP trickling to trickle 800 bytes of content for every 2 megabytes scanned, and to initiate trickling when the HTTP file is 6 megabytes or larger:

```
set av http trickling 6 2 800
```

Example: The following commands enable WebMail scanning only and create a URL pattern named URL_Acme.

- URL host name "www.acme.com"
- path "/acme/marketing"

```
set av http webmail enable
set av http webmail url-pattern-name URL_Acme host www.acme.com
set av http webmail url-pattern-name URL_Acme path /acme/marketing
```

scan-mgr

```
exec av scan-mgr { ... }
get av scan-mgr
set av scan-mgr { ... }
unset av scan-mgr { ... }
```

- scan-mgr** Configures, displays, or performs actions on parameters that control internal AV scanning.
- **content** { **http** | **smtp** | **pop3** } Specifies the types of protocols to examine for virus patterns.
 - **timeout** *number* Specifies a time interval (expressed in seconds) during which the internal AV scanner can complete an initiated virus scan. The scan terminates if the time interval expires before completion. The default setting is 180 seconds.

- **decompress-layer** *number* Specifies how many layers of nested compressed files the internal AV scanner can decompress before it executes the virus scan. For example, if a message contains a compressed .zip file which contains another compressed .zip file, there are two compression layers, and decompressing both files requires a **decompress-layer** setting of 2. Valid settings are between 1 and 20, so the AV scanner can decompress up to 20 layers of compressed files.
- **max-content-size** *number* Specifies the maximum size (in megabytes) of content for a single message that the internal AV scanner scans for virus patterns. If you enable the **drop** switch, and the total content of an incoming message exceeds the maximum, the NetScreen device drops the message content. The range for **max-content-size** is 4 to 20 megabytes, inclusive.
- **max-msgs** *number* Specifies the maximum number of concurrent messages that the internal AV scanner can scan for virus patterns. If you enable the **drop** switch, and the number of messages exceeds the maximum, the internal AV scanner drops the latest message content. The range for **max-msgs** is between 1 and 8 messages, inclusive.
- **pattern-download** Retrieves an updated pattern file directly from a host. This is used for manual updates only.
 - The **tftp-host** parameter specifies the host from which the NetScreen device retrieves an updated pattern file.
 - The **file** parameter specifies the name of the pattern file retrieved from a host.
 - The **version** parameter specifies the version number of the pattern file. This is required to verify that the pattern file is valid.
- **pattern-update-url** *url_str* Specifies the URL address of the server from which the NetScreen device retrieves updates to its pattern file. The value of the URL address should be in the format <http://<host>[:port]/server.ini].

The **interval** *number* setting specifies the time interval (in minutes) during which the NetScreen device starts an automatic update process.

server-name

```
set av name_str server-name name_str [ port port_num ]  
unset av name_str server-name name_str [ port ]
```

server-name Sets the IP address or DNS name for an external antivirus (AV) scanner. **port** *port_num* specifies the port number that Content Scanning Protocol (CSP) uses for communication between the NetScreen device and a Trend Micro InterScan VirusWall server. The device performs this operation on a per-AV-object basis. The default CSP port number is 3300.

BGP Commands

Description: Use the **bgp** context to configure the BGP routing protocol in a NetScreen virtual router.

Context Initiation

Initiating the **bgp** context requires two steps:

1. Enter the **vrouter** context by executing the **set vrouter** command:

```
set vrouter vrouter
```

where *vrouter* is the name of the virtual router. (For all examples that follow, assume that *vrouter* is the trust-vr virtual router.)

2. Enter the **bgp** context by executing the **set protocol bgp** command.

```
ns(trust-vr)-> set protocol bgp id_num
```

where *id_num* is the number of the autonomous system in which the BGP routing instance resides. Once you define an autonomous system number for the BGP routing instance, you no longer have to enter the number in the **set protocol bgp** command.

BGP Commands

The following commands are executable in the **bgp** context.

advertise-def-route

Use the **advertise-def-route** commands to advertise or display the default route in the current virtual router to peers.

Command options: **set**, **unset**

aggregate

Use **aggregate** commands to create, display, or delete aggregate addresses.

Aggregation is a technique for summarizing a range of routing addresses into a single route entry, expressed as an IP address and a subnet mask. Aggregates can reduce the size of the routing table, while maintaining its level of connectivity. In addition, aggregates can reduce the number of advertised addresses, thus reducing overhead.

Command options: **get**, **set**, **unset**

always-compare-med	<p>Use the always-compare-med commands to enable or disable the NetScreen device from comparing paths from each autonomous system (AS) using the Multi-Exit Discriminator (MED). The MED value is one of the criteria that determines the most suitable route to the neighbor device.</p> <p>Command options: get, set, unset</p>
as-number	<p>Use the as-number command to display the autonomous system number configured for the BGP routing instance. When you create the BGP routing instance in a virtual router, you must specify the autonomous system (AS) in which it resides.</p> <p>Command options: get</p>
as-path-access-list	<p>Use as-path-access-list commands to create, remove, or display a regular expression in an AS-Path access list.</p> <p>An AS-path access list serves as a packet filtering mechanism. The NetScreen device can consult such a list and permit or deny BGP packets based on the regular expressions contained in the list. The system can have up to 99 AS-path access lists.</p> <p>Command options: get, set, unset</p>
community-list	<p>Use community-list commands to enter a route in a community list, to remove a route from the list, or to display the list.</p> <p>Command options: get, set, unset</p>
confederation	<p>Use the confederation commands to create a confederation, to remove a confederation, or to display confederation information.</p> <p>Confederation is a technique for dividing an AS into smaller sub-ASs and grouping them. Using confederations reduces the number of connections inside an AS, thus simplifying full mesh topology.</p> <p>Command options: get, set, unset</p>
enable	<p>Use the enable commands to enable or disable the BGP routing protocol in a virtual router.</p> <p>Command options: get, set, unset</p>
flap-damping	<p>Use the flap-damping commands to enable or disable the flap-damping setting.</p> <p>Enabling this setting blocks the advertisement of a route until the route becomes stable. Flap damping allows the NetScreen device to prevent routing instability at an AS border router, adjacent to the region where instability occurs.</p> <p>Command options: get, set, unset</p>

hold-time	<p>Use the hold-time commands to specify or display the maximum amount of time (in seconds) that can elapse between keepalive messages received from the BGP neighbor.</p> <p>Command options: get, set, unset</p>
keepalive	<p>Use the keepalive commands to specify the amount of time (in seconds) that elapses between keepalive packet transmissions. These transmissions ensure that the TCP connection between the local BGP router and a neighbor router stays up.</p> <p>Command options: get, set, unset</p>
local-pref	<p>Use this command to configure a LOCAL_PREF value for the BGP routing protocol. The LOCAL_PREF attribute is the metric most often used in practice to express preferences for one set of paths over another for IBGP.</p> <p>Command options: get, set, unset</p>
med	<p>Use the med commands to specify or display the local Multi-Exit Discriminator (MED).</p> <p>Command options: get, set, unset</p>
neighbor	<p>Use the neighbor commands to set or display configuration parameters for communicating with BGP peers.</p> <p>Command options: clear, exec, get, set, unset</p>
network	<p>Use the network commands to create, display, or delete network and subnet entries. The BGP virtual router advertises these entries to peer devices, without first requiring redistribution into BGP (as with static routing table entries).</p> <p>Command options: get, set, unset</p>
redistribute	<p>Use the redistribute commands to import routes advertised by external routers that use protocols other than BGP, or to display the current redistribution settings.</p> <p>Command options: get, set, unset</p>
reflector	<p>Use the reflector commands to allow the local BGP virtual router to serve as a route reflector. A <i>route reflector</i> is a router that passes Interior BGP (IBGP) learned routes to specified IBGP neighbors (<i>clients</i>), thus eliminating the need for each router in a full mesh to talk to every other router. The clients use the route reflector to readvertise routes to the entire autonomous system (AS).</p> <p>Command options: get, set, unset</p>

reject-default-route

Use the **reject-default-route** commands to enable, disable, or display the **reject-default-route** setting. Enabling this setting makes the NetScreen device ignore default route advertisements from a BGP peer router.

Command options: **get, set, unset**

retry-time

Use the **retry-time** command to specify the amount of time (in seconds) after failing to establish a BGP session with a peer that the local BGP routing instance retries to initiate the session.

synchronization

Use the **synchronization** command to enable synchronization with Interior Gateway Protocol (IGP).

Command options: **set, unset**

advertise-def-route

Description: Use the **advertise-def-route** commands to advertise or display the default route in the current virtual router to BGP peers.

Before you can execute a **advertise-def-route** command, you must initiate the **bgp** context. (See ["Context Initiation" on page 61.](#))

Syntax

set

set advertise-def-route

unset

unset advertise-def-route

Keywords and Variables

None.

aggregate

Description: Use **aggregate** commands to create, display, or delete aggregate addresses.

Aggregation is a technique for summarizing a range of routing addresses into a single route entry. Each aggregate is an address range expressed as an IP address and a subnet mask value. Aggregation can reduce the size of a router's routing table, while maintaining its level of connectivity. In addition, aggregation can reduce the number of advertised addresses, thus reducing overhead.

Before you can execute an **aggregate** command, you must initiate the **bgp** context. (See ["Context Initiation" on page 61.](#))

Syntax

get

```
get aggregate
```

set

```
set aggregate [ ip ip_addr/mask [ as-set ] [ summary-only ] ]
```

unset

```
unset aggregate [ ip ip_addr/mask ]
```


Keywords and Variables

ip

set aggregate ip *ip_addr/mask*

ip Specifies the IP address (*ip_addr*) and subnet mask (*mask*) that comprise the new aggregate address.

Example: The following command creates an aggregate route entry in the trust-vr local router:

```
set aggregate ip 1.1.10.0/24
```

as-set

set aggregate ip *ip_addr/mask* **as-set** [...]

as-set Specifies that the aggregate uses an unordered set of AS numbers (the AS-Set field is set in the AS-Path path attribute) instead of an ordered sequence (the AS-Sequence field is set in the AS-Path path attribute). This option supports the aggregation of routes with different AS-Paths.

summary-only

set aggregate ip *ip_addr/mask* [**as-set**] **summary-only**

summary-only Suppresses more specific routes in the routing table.

Example: The following command specifies that the aggregate uses an unordered set of AS numbers, while suppressing more specific routes.

```
set aggregate ip 3.3.3.3/24 as-set summary-only
```

always-compare-med

Description: Use the **always-compare-med** commands to enable or disable the NetScreen device from comparing paths from each autonomous system (AS) using the Multi-Exit Discriminator (MED). The MED is one of the criteria that determines the most suitable route to the neighbor device.

Before you can execute an **always-compare-med** command, you must initiate the **bgp** context. (See ["Context Initiation" on page 61.](#))

Syntax

get

`get always-compare-med`

set

`set always-compare-med`

unset

`unset always-compare-med`

Keywords and Variables

None.

as-number

Description: Use the **as-number** command to display the autonomous system number configured for the BGP routing instance. When you create the BGP routing instance in a virtual router, you must specify the autonomous system (AS) in which it resides.

Before you can execute the **as-number** command, you must initiate the **bgp** context. (See ["Context Initiation" on page 61.](#))

Syntax

get

get as-number

Keywords and Variables

None.

as-path-access-list

Description: Use **as-path-access-list** commands to create, remove, or display a regular expression in an AS-Path access list.

An AS-path access list serves as a packet filtering mechanism. The NetScreen device can consult such a list and permit or deny BGP packets based on the regular expressions contained in the list.

Before you can execute an **as-path-access-list** command, you must initiate the **bgp** context. (See ["Context Initiation" on page 61.](#))

Syntax

get

```
get as-path-access-list
```

set

```
set as-path-access-list id_num { deny | permit } string
```

unset

```
unset as-path-access-list id_num { deny | permit } string
```

Keywords and Variables

Variable Parameters

```
set as-path-access-list id_num { deny | permit } string  
unset as-path-access-list id_num { deny | permit } string
```

<i>id_num</i>	The identification number of the access list (range 1 - 99 inclusive).
<i>string</i>	<p>The regular expression used for BGP packet filtering. You can use the following in the regular expression:</p> <ul style="list-style-type: none">• '^' The start of a path• '\$' The end of a path• '{' The start of an AS_SET• '}' The end of an AS_SET• '(' The start of an AS_CONFED_SET or AS_CONFED_SEQ• ')' The end of an AS_CONFED_SET or AS_CONFED_SEQ• '.' Matches any single character• '*' Matches zero or more characters• '+' Matches one or more characters• '_' Matches zero or one instance of a punctuation character• '[' Specifies a set of characters• '-' Used within brackets to specify a range of AS numbers• '^' Used as the first item within brackets to exclude AS numbers

deny | permit

```
set as-path-access-list id_num { deny | permit } string  
unset as-path-access-list id_num { deny | permit } string
```

deny | permit Denies or permits BGP packets containing the regular expression (*string*).

Example: The following command places the regular expression "23" in an AS-Path access list with ID number 10:

```
set as-path-access-list 10 permit 23
```

community-list

Description: Use **community-list** commands to create a community list that defines community attributes of routes that are permitted or denied.

A community consists of routes that are associated with the same identifier. Routers can use the community identifier when they need to treat two or more advertised routes in the same way.

Before you can execute a **community-list** command, you must initiate the **bgp** context. (See ["Context Initiation" on page 61.](#))

Syntax

get

```
get community-list
```

set

```
set community-list id_num1
{
  default-permit |
  deny | permit
}
[
  number |
  as id_num2 id_num3 |
  no-advertise |
  no-export |
  no-export-subconfed
]
```

unset

```
unset community-list id_num1
{
  default-permit |
  deny | permit
}
[
  number |
  as id_num2 id_num3 |
  no-advertise |
  no-export |
  no-export-subconfed
]
```

Keywords and Variables

Variable Parameters

```
set community-list id_num1 { deny | permit | default-permit } number
unset community-list id_num1 { deny | permit | default-permit } number
```

id_num1 The identifier of the community list (range 1 - 99 inclusive).

number The community number, which can be between 0-65535 inclusive.

Example: The following command defines the community list 20 that denies routes with the community value 200.

```
set community-list 20 deny 200
```


as

```
set community-list id_num1 { deny | permit } as id_num2 id_num3
unset community-list id_num1 { deny | permit } as id_num2 id_num3
```

as Defines a private community, in the form of an AS number (*id_num2*) and a community number defined within the AS (*id_num3*). The community number can be between 0-65535 inclusive.

Example: The following command creates a community list with an ID of 10 that permits the community 11 in AS 10000:

```
set community-list 10 permit as 10000 11
```

deny | permit | default-permit

```
set community-list id_num1 { deny | permit } [ ... ]
unset community-list id_num1 { deny | permit } [ ... ]
```

deny | permit Denies or permits routes with the specified community value.

default-permit Permits the route if it does not match any community value specified in the community list. By default, routes that do not match community values in the community list are denied.

Example: The following command defines the community list 20 that denies routes with the community value 200.

```
set community-list 20 deny 200
```

no-advertise

```
set community-list id_num1 { deny | permit } no-advertise
set community-list id_num1 { deny | permit } no-advertise
```

no-advertise Specifies that the NetScreen device does not advertise routes with this community value in the communities attribute to any peer devices.

no-export

```
set community-list id_num1 { deny | permit } no-export  
set community-list id_num1 { deny | permit } no-export
```

no-export Specifies that the NetScreen device does not advertise routes with this community value to EBGp peers, except sub-autonomous systems within the confederation.

no-export-subconfed

```
set community-list id_num1 { deny | permit } no-export-subconfed  
set community-list id_num1 { deny | permit } no-export-subconfed
```

no-export-subconfed Specifies that the NetScreen device does not advertise routes with this community value to any external peers.

confederation

Description: Use the **confederation** commands to create a confederation, to remove a confederation, or to display confederation information.

Confederation is a technique for dividing an AS into smaller sub-ASs and grouping them. Using confederations reduces the number of connections inside an AS, simplifying the routing matrices created by meshes.

Before you can execute a **confederation** command, you must initiate the **bgp** context. (See ["Context Initiation" on page 61.](#))

Syntax

get

```
get confederation
```

set

```
set confederation { id id_num1 | peer id_num2 | rfc3065 }
```

unset

```
unset confederation { id | peer id_num2 | rfc3065 }
```

Keywords and Variables

id

```
set confederation id id_num1  
unset confederation id
```

id The identification number (*id_num1*) of the confederation.

Example: The following command creates a confederation with an ID of 10:

```
set confederation id 10
```

peer

```
set confederation peer id_num2  
unset confederation peer id_num2
```

peer *id_num2* The identifier of a new peer autonomous system (AS) entry.

Example: The following command adds AS 45040 to the confederation:

```
set confederation peer 45040
```

rfc3065

```
set confederation rfc3065  
unset confederation rfc3065
```

rfc3065 Specifies configuration in compliance with RFC 3065. The default is compliance with RFC 1965.

enable

Description: Use the **enable** commands to enable or disable the BGP routing protocol in a virtual router.

Before you can execute an **enable** command, you must initiate the **bgp** context. (See ["Context Initiation" on page 61.](#))

Syntax

set

set enable

unset

unset enable

Keywords and Variables

None.

flap-damping

Description: Use the **flap-damping** commands to enable or disable the **flap-damping** setting.

Enabling this setting blocks the advertisement of a route until the route becomes stable. Flap damping allows the NetScreen device to contain routing instability at an AS border router, adjacent to the region where instability occurs.

Before you can execute a **flap-damping** command, you must initiate the **bgp** context. (See ["Context Initiation" on page 61.](#))

Syntax

set

set flap-damping

unset

unset flap-damping

Keywords and Variables

None.

hold-time

Description: Use the **hold-time** commands to specify or display the maximum amount of time (in seconds) that can elapse between keepalive messages received from the BGP neighbor. If the hold-time elapses before any message is received from a BGP neighbor, the session is considered down. The default is 180 seconds.

Note: The default **keepalive** value is always one third of the current **hold-time** value.

Before you can execute a **hold-time** command, you must initiate the **bgp** context. (See ["Context Initiation" on page 61.](#))

Syntax

get

```
get hold-time
```

set

```
set hold-time number
```

unset

```
unset hold-time
```

Keywords and Variables

Variable Parameter

```
set hold-time number
```

number The maximum length of time (in seconds) between messages.

keepalive

Description: Use the **keepalive** commands to specify the amount of time (in seconds) that elapses between keepalive packet transmissions. These transmissions ensure that the TCP connection between the local BGP router and a neighbor router stays up. The default value is one-third of the hold-time value (for the default **hold-time** value of 180 seconds, the default **keepalive** value is 60 seconds).

Before you can execute a **keepalive** command, you must initiate the **bgp** context. (See ["Context Initiation" on page 61.](#))

Syntax

get

`get keepalive`

set

`set keepalive number`

unset

`unset keepalive`

Keywords and Variables

Variable Parameter

<i>number</i>	The maximum length of time (in seconds) between keepalive messages.
---------------	---

local-pref

Description: Use the **local-pref** commands to configure the Local-Pref path attribute for the BGP routing protocol.

The Local-Pref path attribute is a metric used to inform IBGP peers of the local router's preference for the route. The higher the value, the greater the preference. Routers advertise this attribute to internal peers (peers in the same AS) and to neighboring confederations, but never to external peers. The default value is 100.

Before you can execute a **local-pref** command, you must initiate the **bgp** context. (See ["Context Initiation" on page 61.](#))

Syntax

get

get local-pref

set

set local-pref *number*

unset

unset local-pref

Keywords and Variables

Variable Parameter

set local-pref *number*

number The preference level for the virtual router.

med

Description: Use the **med** commands to specify or display the local Multi-Exit Discriminator (MED).

The MED is an attribute that notifies a neighbor in another AS of the optimal path to use when there are multiple entry points to the AS. If an EBGp update contains a MED value, the BGP routing instance sends the MED to all IBGP peers within the AS. If you assign a MED value, this value overrides any MED values received in update messages from external peers.

Note that although you set the MED in the local AS, the neighbor in another AS uses the MED value to decide which entry point to use. If all other factors are equal, the path with the lowest MED value is chosen. The default MED value is 0.

Before you can execute a **med** command, you must initiate the **bgp** context. (See ["Context Initiation" on page 61.](#))

Syntax

get

get med

set

set med *id_num*

unset

unset med

Keywords and Variables

Variable Parameter

set med *id_num*

id_num The identification number of the MED.

Example: The following command specifies MED 100 for the virtual router trust-vr:

set med 100

neighbor

Description: Use the **neighbor** commands to set or display general configuration parameters for communicating with BGP peers.

Before you can execute a **neighbor** command, you must initiate the **bgp** context. (See ["Context Initiation" on page 61.](#))

Syntax

clear

```
clear neighbor ip_addr1
{
  flap-route ip_addr2 [ add ] |
  stats
}
```

exec

```
exec neighbor ip_addr
{
  connect |
  disconnect |
  tcp-connect
}
```

get

```
get neighbor [ ip_addr | peer-group name_str ]
```

set

```
set neighbor
{
  ip_addr
  [
    advertise-def-route |
    ebgp-multihop number |
    enable |
    force-reconnect |
    hold-time number |
    keepalive number |
    md5-authentication string |
    med number |
    nhself-enable |
    peer-group name_str |
    reflector-client |
    reject-default-route |
    remote-as number
    [
      local-ip ip_addr |
      outgoing-interface interface |
      src-interface interface
    ] |
    retry-time number |
    route-map name_str { in | out } |
    send-community |
    weight number
  ] |
  peer-group name_str
  [
    ebgp-multihop number |
    force-reconnect |
    hold-time number |
    keepalive number |
```

```
    md5-authentication string |
    nhself-enable |
    reflector-client |
    remote-as number [ local-ip ip_addr ] |
    retry-time number |
    route-map name_str { in | out } |
    send-community |
    weight number
  ]
}
```

unset

```
unset neighbor
{
  ip_addr
  [
    advertise-def-route |
    ebgp-multihop |
    enable |
    force-reconnect |
    hold-time |
    keepalive |
    md5-authentication string |
    med |
    nhself-enable |
    peer-group |
    reflector-client |
    reject-default-route |
    remote-as number
    [
      local-ip ip_addr |
      outgoing-interface interface |
      src-interface interface
    ] |
  ]
}
```

```
    retry-time |
    route-map name_str { in | out } |
    send-community |
    weight
  ]
peer-group name_str
[
  ebgp-multihop |
  force-reconnect |
  hold-time |
  keepalive |
  md5-authentication string |
  nhself-enable |
  reflector-client |
  remote-as number [ local-ip ip_addr ] |
  retry-time |
  route-map name_str { in | out } |
  send-community |
  weight
]
}
```

Keywords and Variables

Variable Parameter

```
clear neighbor ip_addr  
get neighbor ip_addr  
set neighbor ip_addr { ... }  
unset neighbor ip_addr { ... }
```

ip_addr The IP address of the neighboring peer device.

Example: The following command displays information about a neighbor device at IP address 1.1.100.101:

```
get neighbor 1.1.100.101
```

advertise-def-route

```
set neighbor ip_addr advertise-def-route  
unset neighbor ip_addr advertise-def-route
```

advertise-def-route Advertises the default route in the current virtual router to the BGP peer.

connect

```
exec neighbor ip_addr connect
```

connect Establishes a BGP connection to the neighbor. You can use this command for troubleshooting a BGP connection.

disconnect

```
exec neighbor ip_addr disconnect
```

disconnect Terminates the BGP connection to the neighbor. You can use this command for troubleshooting a BGP connection.

ebgp-multihop

```
set neighbor { ip_addr | peer-group name_str } ebgp-multihop number  
unset neighbor { ip_addr | peer-group name_str } ebgp-multihop
```

ebgp-multihop The number of intervening routing nodes (*number*) allowed between the local BGP router and the BGP neighbor (*ip_addr*). A setting of zero (the default value) disables the multihop feature. The local BGP router uses the **ebgp-multihop** value as TTL in all IP packets transmitted to the neighbor.

Example: The following command directs the virtual router to allow three intervening route nodes between the virtual router and a neighbor device at IP address 1.1.100.101:

```
set neighbor 1.1.100.101 ebgp-multihop 3
```

enable

```
set neighbor ip_addr enable  
unset neighbor ip_addr enable
```

enable Enables or disables peer communications.

force-reconnect

```
set neighbor { ip_addr | peer-group name_str } force-reconnect
unset neighbor { ip_addr | peer-group name_str } force-reconnect
```

force-reconnect Causes the peer to drop the existing BGP connection and accept a new connection. You can use this option when NSRP failover occurs but the failover interval is long enough that the BGP peer still considers the connection to be active and rejects new connection attempts.

hold-time

```
set neighbor { ip_addr | peer-group name_str } hold-time number
unset neighbor { ip_addr | peer-group name_str } hold-time
```

hold-time Specifies the number of seconds (*number*) that the current BGP speaker waits to receive a message from its neighbor. The default is 180 seconds.

Example: The following command specifies a hold-time value of 60:

```
set neighbor 1.1.10.10 hold-time 60
```

keepalive

```
set neighbor { ip_addr | peer-group name_str } keepalive number
unset neighbor { ip_addr | peer-group name_str } keepalive
```

keepalive Specifies the maximum amount of time (in seconds) that can elapse between keepalive packet transmissions before the local BGP virtual router terminates the connection to the neighbor. The default is one-third of the hold-time value (for the default **hold-time** value of 180 seconds, the default **keepalive** value is 60 seconds).

Example: The following command specifies a keepalive value of 90 seconds:

```
ns(trust-vr/bgp)-> set neighbor 1.1.100.101 keepalive 90
```

md5-authentication

```
set neighbor { ip_addr | peer-group name_str } md5-authentication string
unset neighbor { ip_addr | peer-group name_str } md5-authentication string
```

md5-authentication Specifies the BGP peer MD5 authentication string. The maximum length is 32 characters.

Example: The following command specifies an MD5 authentication string (5784ldk094):

```
set neighbor 1.1.100.101 md5-authentication 5784ldk094
```

med

```
set neighbor ip_addr med id_num
unset neighbor ip_addr med
```

med Specifies the ID number (*id_num*) of the local Multi-Exit Discriminator (MED). The default value is 0.

Example: The following command specifies the Multi-Exit Discriminator (MED) **20099** for a neighbor with IP address **1.1.10.10**:

```
set neighbor 1.1.10.10 med 20099
```

nhself-enable

```
set neighbor { ip_addr | peer-group name_str } nhself-enable
unset neighbor { ip_addr | peer-group name_str } nhself-enable
```

nhself-enable Specifies that the Next-Hop path attribute for routes sent to this peer is set to the interface IP address of the local virtual router.

Example: The following command makes the local virtual router the next hop value for the peer 1.1.10.10:

```
set neighbor 1.1.10.10 nhself-enable
```

peer-group

```
get neighbor peer-group name_str
set neighbor ip_addr peer-group name_str [ ... ]
set neighbor peer-group name_str [ ... ]
unset neighbor ip_addr peer-group name_str [ ... ]
unset neighbor peer-group name_str [ ... ]
```

peer-group The name of a group of BGP neighbors. Each BGP neighbor in a peer group shares the same update policies. This allows you to set up policies that apply to all the BGP peers instead of creating a separate policy for each peer. Use this command to both create the peer-group and configure peer-group parameters.

reflector-client

```
set neighbor { ip_addr | peer-group name_str } reflector-client
unset neighbor { ip_addr | peer-group name_str } reflector-client
```

reflector-client Specifies that the neighbor is a reflector client in the route reflector cluster. The local BGP routing instance is the route reflector.

Example: The following command specifies that the neighbors in the peer group Acme_Peers are reflector clients:

```
set neighbor peer-group Acme_Peers reflector-client
```

reject-default-route

```
set neighbor ip_addr reject-default-route
unset neighbor ip_addr reject-default-route
```

reject-default-route Specifies that the local BGP routing instance is to ignore default route advertisements from the peer. By default, default routes advertised by peers are added to the local routing table.

remote-as

```
set neighbor { ip_addr | peer-group name_str }  
    remote-as number [ local-ip ip_addr ]  
set neighbor { ip_addr | peer-group name_str }  
    remote-as number ( outgoing-interface interface | src-interface interface )  
unset neighbor { ip_addr | peer-group name_str }  
    remote-as number [ local-ip ip_addr ]
```

remote-as Identifies the remote AS (*number*) to be the neighbor of the current BGP speaker.
local-ip *ip_addr* specifies the local IP address for EBGp multi-hop peer.
outgoing-interface *interface* specifies the outgoing interface to which BGP binds.
src-interface *interface* specifies the source interface to which the BGP binds.

Example: The following command identifies AS 30 as the remote AS for the peer 1.1.10.10:

```
set neighbor 1.1.10.10 remote-as 30
```

retry-time

```
set neighbor { ip_addr | peer-group name_str } retry-time number  
unset neighbor { ip_addr | peer-group name_str } retry-time number
```

retry-time Specifies the time (in seconds) that the BGP routing instance retries to establish a session with the peer after an unsuccessful BGP session establishment attempt. The default is 120 seconds.

route-map

```
set neighbor { ip_addr | peer-group name_str } route-map name_str { in | out }  
unset neighbor { ip_addr | peer-group name_str } route-map name_str { in | out }
```

route-map Specifies the route map to use for the BGP neighbor. The **in** | **out** switches determine if the route map applies to incoming or outgoing routes.

Example: The following command specifies that the route map Mkt_Map applies to incoming routes from the neighbor at IP address 1.1.10.10:

```
set neighbor 1.1.10.10 route-map Mkt_Map in
```

send-community

```
set neighbor { ip_addr | peer-group name_str } send-community  
unset neighbor { ip_addr | peer-group name_str } send-community
```

send-community Directs the BGP routing protocol to transmit the community attribute to the neighbor. By default, the community attribute is not sent to neighbors.

tcp-connect

```
exec neighbor ip_addr tcp-connect
```

tcp-connect Tests the TCP connection to the neighbor. You can use this command for troubleshooting a TCP connection.

weight

```
set neighbor { ip_addr | peer-group name_str } weight number  
unset neighbor { ip_addr | peer-group name_str } weight
```

weight The preference for routes learned from this neighbor. The higher the value, the more preference given to the routes learned from this neighbor. The default value is 100.

Example: The following command assigns a weight of 200 to the path to the neighbor at IP address 1.1.10.10:

```
set neighbor 1.1.10.10 weight 200
```

network

Description: Use the **network** commands to create, display, or delete network and subnet entries that are reachable from the virtual router. BGP advertises these entries to peer devices, without first requiring redistribution into BGP (as with static routing table entries).

Before you can execute a **network** command, you must initiate the **bgp** context. (See ["Context Initiation" on page 61.](#))

Syntax

get

```
get network
```

set

```
set network ip_addr1/mask1 [ check ip_addr2/mask2 | no-check ]
```

unset

```
unset network ip_addr1/mask1 [ check ip_addr2/mask2 | no-check ]
```

Keywords and Variables

Variable Parameters

```
set network ip_addr1/mask1 [ ... ]  
unset network ip_addr1/mask1
```

ip_addr1/mask1 The IP address and subnet mask of the network. The mask does not have to be the same as the subnet mask used in the network. For example, 10.0.0.0/8 is a valid network to be advertised by BGP. When the **check** option is used, *ip_addr1/mask1* can be a MIP address range.

Example: The following command creates a network entry (10.1.0.0/16) for the virtual router **trust-vr**:

```
set network 10.1.0.0/16
```

check

```
set network ip_addr1/mask1 check ip_addr2/mask2
```

check Directs the device to check *ip_addr2/mask2* for network reachability before advertising *ip_addr1/mask1* to BGP peers. If *ip_addr2/mask2* is reachable, BGP advertises *ip_addr1/mask1* to its peers. If *ip_addr2/mask2* becomes unreachable, BGP withdraws the route *ip_addr1/mask1* from its peers.

no-check

```
set network ip_addr1/mask1 no-check
```

no-check Directs the device not to check for network reachability.

redistribute

Description: Use the **redistribute** commands to import routes advertised by external routers that use protocols other than BGP. Use the **get redistribution** command to display current redistribution settings.

Before you can execute a **redistribute** command, you must initiate the **bgp** context. (See ["Context Initiation" on page 61.](#))

Syntax

get

```
get redistribution
```

set

```
set redistribute route-map name_str protocol  
    { connected | imported | ospf | rip | static }
```

unset

```
unset redistribute route-map name_str protocol  
    { connected | imported | ospf | rip | static }
```

Keywords and Variables

protocol

```
set redistribute route-map name_str protocol [ ... ]  
unset redistribute route-map name_str protocol [ ... ]
```

protocol The protocol from which the redistributed routes were learned. This can be one of the following:
connected, imported, ospf, rip, static.

route-map

```
set redistribute route-map name_str protocol [ ... ]  
unset redistribute route-map name_str protocol [ ... ]
```

route-map The name (*name_str*) of the route map to be used to filter routes.

reflector

Description: Use the **reflector** commands to allow the local virtual router to serve as a route reflector to clients in a cluster.

A *route reflector* is a router that passes Interior BGP (IBGP) learned routes to specified IBGP neighbors (*clients*), thus eliminating the need for each router in a full mesh to talk to every other router. A cluster consists of multiple routers, with a single router designated as the route reflector, and the others as clients. Routers outside of the cluster treat the entire cluster as a single entity, instead of interfacing with each individual router in full mesh. This arrangement greatly reduces overhead. The clients exchange routes with the route reflector, while the route reflector reflects routes between clients.

To configure clients in the cluster, use the [reflector-client](#) option of the [neighbor](#) commands.

Before you can execute a **reflector** command, you must initiate the **bgp** context. (See ["Context Initiation" on page 61.](#))

Syntax

get

```
get reflector
```

set

```
set reflector [ cluster-id id_num ]
```

unset

```
unset reflector [ cluster-id id_num ]
```

Keywords and Variables

cluster-id

```
set reflector cluster-id id_num  
unset reflector cluster-id id_num
```

cluster-id The ID number (*id_num*) of the cluster. The cluster ID allows the BGP routing instance to append the cluster ID to the cluster list of a route. BGP must be disabled before you can set the cluster ID.

Example: The following command allows the local BGP routing instance to serve as a route reflector, and sets the cluster ID to 20:

```
set reflector  
set reflector cluster-id 20
```

reject-default-route

Description: Use the **reject-default-route** commands to enable, disable, or display the **reject-default-route** setting. Enabling this setting makes the NetScreen device ignore default route advertisements from a BGP peer router. By default, BGP accepts default routes advertised by BGP peers.

Before you can execute an **reject-default-route** command, you must initiate the **bgp** context. (See ["Context Initiation" on page 61.](#))

Syntax

get

`get reject-default-route`

set

`set reject-default-route`

unset

`unset reject-default-route`

Keywords and Variables

None.

retry-time

Description: Use the **retry-time** command to specify the amount of time (in seconds) after failing to establish a BGP session with a peer that the local BGP routing instance retries to initiate the session. The default is 120 seconds.

Before you can execute a **retry-time** command, you must initiate the **bgp** context. (See ["Context Initiation" on page 61.](#))

Syntax

set

set retry-time *number*

unset

unset retry-time

Keywords and Variables

None.

synchronization

Description: Use the **synchronization** command to enable synchronization with an Interior Gateway Protocol (IGP), such as OSPF.

If an EBGp router advertises a route before other routers in the AS learn the route via an IGP, traffic forwarded within the AS could be dropped if it reaches a router that has not learned the route. Synchronization prevents this from occurring by ensuring that a BGP router does not advertise a route until it has also learned the route through an IGP.

Before you can execute a **synchronization** command, you must initiate the **bgp** context. (See [“Context Initiation” on page 61.](#))

Syntax

set

set synchronization

unset

unset synchronization

Keywords and Variables

None.

chassis

Description: Use the **set chassis** command to activate the audible alarm feature, or to set the normal and severe temperature thresholds for triggering temperature alarms.

Syntax

get

```
get chassis
```

set

```
set chassis
{
  audible-alarm
    { all | battery | fan-failed | power-failed | temperature } |
  temperature-threshold
    { alarm | severe }
      { celsius number | fahrenheit number }
```

unset

```
unset chassis
{
  audible-alarm
    { all | battery | fan-failed | power-failed | temperature } |
  temperature-threshold
    { alarm | severe }
      { celsius number | fahrenheit number }
```


Arguments

audible-alarm	Enables or disables the audible alarm to announce hardware failure events. <ul style="list-style-type: none">• all Enables or disables the audible alarm in the event of a fan failure, a interface module failure, a power supply failure, or a temperature increase above an admin-defined threshold.• battery Enables or disables the audible alarm in the event of battery failure.• fan-failed Enables or disables the audible alarm in the event of a fan failure.• module-failed Enables or disables the audible alarm in the event of an interface module failure.• power-failed Enables or disables the audible alarm in the event of a power supply failure.• temperature Enables or disables the audible alarm if the temperature rises above an admin-defined threshold.
temperature-threshold	Defines the temperature (celsius or fahrenheit) required to trigger a regular alarm or a severe alarm. A severe alarm sounds a greater frequency of audible alarms and generates a greater number of event log entries.

Example: To enable the audible alarm to sound in the event that one or more of the fans in the fan assembly fails:

```
set chassis audible-alarm fan-failed
```

clock

Description: Use the **clock** commands to set the system time on the NetScreen device.

Note: By default, the NetScreen device automatically adjusts its system clock for daylight saving time.

Syntax

get

```
get clock
```

set

```
set clock { date [ time ] | dst-off | ntp | timezone number }
```

unset

```
unset clock { dst-off | ntp | timezone }
```

Keywords and Variables

Variable Parameter

```
set clock date time
```

date time Configures the correct current date and time on the NetScreen device. Specify the date and time using the following formats: (*mm/dd/yyyy hh:mm* or *mm/dd/yyyy hh:mm:ss*).

Example: The following command sets the clock to December 15, 2002, 11:00am:

```
set clock 12/15/2002 11:00
```

dst-off

```
set clock dst-off  
unset clock dst-off
```

dst-off Turns off the automatic time adjustment for daylight saving time.

ntp

```
set clock ntp  
unset clock ntp
```

ntp Configures the device for Network Time Protocol (NTP), which synchronizes computer clocks in the Internet.

timezone

```
set clock timezone number  
unset clock timezone number
```

timezone Sets the current time zone value. This value indicates the time difference between GMT standard time and the current local time (when DST is OFF). When DST is ON and the clock is already set forward one hour, decrease the time difference by one hour and set the minutes accurately. Set the number between -12 and 12.

common-criteria

Description: Use the **common-criteria** command to disable all internal commands. Only the root admin can set this command. If someone other than the root admin tries to set this command, the NetScreen device displays an error message.

Syntax

set

```
set common-criteria no-internal-commands
```

unset

```
unset common-criteria no-internal-commands
```

Keywords and Variables

no-internal-commands

```
set common-criteria no-internal-commands  
unset common-criteria no-internal-commands
```

no internal commands Disables all internal commands.

“config” through “interface”

This section lists and describes Juniper Networks NetScreen Command Line Interface (CLI) commands **config** through **interface**.

Note: As you execute CLI commands using the syntax descriptions in this section, you may find that certain commands and command features are unavailable on your NetScreen platform. A good example is the **vsys** command, which is available on a NetScreen-500 device, but not on a NetScreen-5XP device. Similarly, some command options are unavailable on certain platforms, as with the **df-bit** option of the **vpn** command. This option is available on a NetScreen-500, but not on a NetScreen-5XP.

config

Description: Use the **config** command to display the configuration settings for a NetScreen device or interface. You can display recent configuration settings (stored in RAM), or saved configurations (stored in flash memory).

Syntax

exec

```
exec config { lock { abort | end | start } | rollback [ enable | disable ] }
```

get

```
get config [ all | datafile | lock | rollback | saved ]
```

set

```
set config lock timeout number
```

unset

```
unset config lock timeout
```

Keywords and Variables

all

get config all

all Displays all configuration information.

datafile

get config datafile

datafile Displays the Security Manager datafile, which resides on the NetScreen device and contains current device configurations formatted according to the Security Manager syntax schema. ScreenOS generates the datafile from the current device configuration when the Security Manager management system queries the device.

lock

exec config lock start
exec config lock end
exec config lock abort
set config lock timeout *number*
unset config lock timeout

lock Instructs the NetScreen device to lock a configuration file in memory for a specified time interval.

- **exec config lock** Locks/unlocks the configuration file in memory. You can also abort the lockout and immediately restart the device with the configuration file that was previously locked in memory.
- **set config lock timeout** Changes the default lockout period, which is five minutes.

rollback

```
exec config rollback
exec config rollback enable
exec config rollback disable
get config rollback
```

rollback

- **exec config rollback**

Reverts the NetScreen device to the LKG (last-known-good) configuration—providing that a LKG configuration is available.

- **exec config rollback**

- **enable** Enables the NetScreen device to automatically rollback to the LKG configuration in case of a problem when loading a new configuration.
- **disable** Disables the automation of the configuration rollback feature on the NetScreen device. If you disable the automation of this feature, you can still perform a configuration rollback manually using the **exec config rollback** command.

- **get config rollback**

Indicates if a LKG configuration is available for configuration rollback and also indicates if the automatic config rollback feature is enabled.

If there is a LKG configuration saved in memory, the output of the command displays:

"\$lkg\$.cfg" (the name of the LKG file).

The config rollback feature is enabled if the output of the command displays "= yes" at the end of the string. For example:

" "\$lkg\$.cfg" " = yes"

If the feature is not enabled, the output displays a blank space instead of "yes".

saved

```
get config saved
```


console

Description: Use the **console** commands to define or list the CLI console parameters.

The console parameters determine the following:

- Whether the NetScreen device displays messages in the active console window
- The number of lines that may appear on a console window page
- The maximum time that can pass before automatic logout occurs due to inactivity

If console access is currently disabled, you can enable it using the **unset console disable** command through a Telnet connection.

Syntax

get

```
get console
```

set

```
set console  
{  
  aux disable |  
  disable |  
  page number |  
  timeout number  
}
```

unset

```
unset console { aux disable | disable | page | timeout }
```

Keywords and Variables

aux disable

```
set console aux disable
unset console aux disable
```

aux disable Enables or disables the auxiliary modem console port. Some platforms have this auxiliary port, in addition to the standard console port. An admin can use the auxiliary modem console port to execute CLI configuration commands. Use the **aux disable** switch to disable the port, when you need to enforce strict security by excluding admin access through this port.

disable

```
set console disable
unset console disable
```

disable Disables console access through the serial port. Two confirmations are required to disable access to the console. Executing this option saves the current NetScreen configuration and closes the current login session. **Note:** After you execute the **console disable** option, non-serial console sessions can still function (as with SSH and Telnet).

page

```
set console page number
unset console page
```

page An integer value specifying how many lines appear on each page between page breaks. When you set this value to zero, there are no page breaks, and the text appears in a continual stream.

Example: To define 20 lines per page displayed on the console:

```
set console page 20
```

timeout

```
set console timeout number  
unset console timeout
```

timeout Determines how many minutes the device waits before closing an inactive administrator session. If you set the value to zero, the console never times out.

Example: To define the console timeout value to 40 minutes:

```
set console timeout 40
```

Defaults

Access to the serial console is enabled.

The console displays 22 lines per page.

The default inactivity timeout is 10 minutes.

The NetScreen device sends console messages to the buffer by default.

counter

Description: Use the **counter** commands to clear or display the values contained in traffic counters.

Traffic counters provide processing information, which you can use to monitor traffic flow. NetScreen devices maintain the following categories of counters:

- Screen counters, for monitoring firewall behavior for the entire zone or for a particular interface
- Policy counters, for reporting the amount of traffic affected by specified policies
- Hardware counters, for monitoring hardware performance and tracking the number of packets containing errors
- Flow counters, for monitoring the number of packets inspected at the flow level

Syntax

clear

```
clear [ cluster ] counter
{
  all |
  ha |
  screen [ interface interface | zone zone ]
}
```

get

```
get counter
{
  flow | statistics
    [ interface interface | zone zone ] |
  screen { interface interface | zone zone }
  policy pol_num { day | hour | minute | month | second }
}
```

Keywords and Variables

cluster

```
clear [ cluster ] counter [ ... ]
```

cluster Propagates the **clear** operation to all other devices in a NSRP cluster.

Example: To clear the contents of all counters and propagate the operation to all devices in the cluster:

```
clear cluster counter all
```

flow

```
get counter flow [ ... ]
```

flow Specifies counters for packets inspected at the flow level. A flow-level inspection examines various aspects of a packet to gauge its nature and intent.

ha

```
clear [ cluster ] counter ha
```

ha Specifies counters for packets transmitted across a high-availability (HA) link between two NetScreen devices. An HA-level inspection keeps count of the number of packets and packet errors.

interface

```
clear [ cluster ] counter screen interface interface
```

interface The name of the interface. Specifies counters for packets inspected at the interface level. The inspection checks for packet errors and monitors the quantity of packets according to established threshold settings. For more information on interfaces, see ["Interface Names" on page A-1](#).

policy

```
get counter policy pol_num { day | hour | minute | month | second }
```

policy Identifies a particular policy (*pol_num*). This allows you to monitor the amount of traffic that the policy permits.

day | hour | minute | month | second Specifies the period of time for monitoring traffic permitted by a particular policy.

screen

```
clear [ cluster ] counter screen [ interface interface | zone zone ]  
get counter screen { interface interface | zone zone }
```

screen Clears the screen counters. The **interface** *interface* parameter specifies the name of a particular interface. For more information on interfaces, see ["Interface Names" on page A-I](#).

statistics

```
get counter statistics [ ... ]
```

statistics Displays the counter statistics.

zone

```
get counter screen zone zone
```

zone Identifies the zone, and specifies counters for packets inspected at the zone level. The inspection checks for packet errors and monitors the quantity of packets according to established threshold settings. For more information on interfaces, see ["Interface Names" on page A-I](#).

delete

Description: Use **delete** to delete persistent information in flash memory.

Syntax

delete

```
delete [ cluster ]
{
  crypto { auth-key | file } |
  file dev_name:filename |
  node_secret ipaddr ip_addr |
  nsmgmt keys |
  pki object-id number |
  ssh device all
}
```

Keywords and Variables

crypto

```
delete [ cluster ] crypto auth-key
delete [ cluster ] crypto file
```

- | | |
|-------------------|--|
| crypto | Removes encrypted items from flash memory. |
| • auth-key | Removes image signature verification key. |
| • file | Remove all crypto hidden files. |

file

delete file *dev_name:filename*

dev_name:filename Deletes the file residing on the module named *dev_name* from the flash card memory.

filename Identifies the file name.

Examples: The following command deletes a file named **myconfig** in the flash memory on the memory board:

```
clear file flash:myconfig
```

node_secret ipaddr

delete node_secret ipaddr *ip_addr*

**node_secret
ipaddr**

Deletes the SecurID stored node secret. The node secret is a 16-byte key shared between the SecurID Ace server and its clients (which may include the NetScreen device). The server and the clients use this key to encrypt exchanged traffic. The Ace Server sends the node secret to the NetScreen device during initial authentication.

The node secret *must* remain consistent with the ACE Server. Otherwise, there can be no communication between the NetScreen device and the ACE Server. You can detect communication problems by checking the ACE Server log for a message saying that the node secret is invalid. If you find such a message, the solution is as follows.

- Execute **delete node_secret**.
- On the ACE Server, change the configuration for the client (the NetScreen device) to say that the server did *not* send the node secret.

This causes the NetScreen device to request the node secret, and authorizes the ACE Server to send a new one. This action resynchronizes communication.

The **ipaddr** *ip_addr* parameter clears the node secret associated with the outgoing IP address of the interface that communicates with the SecurID server (*ip_addr*).

nsmgmt

delete nsmgmt keys

nsmgmt keys Deletes the public and private keys for nsmgmt. The NetScreen device uses these keys to encrypt and decrypt the Configlet file.

pki object-id

delete pki object-id *id_num*

pki object-id Deletes a particular PKI object (*id_num*).

ssh device all

delete ssh device all

ssh device all Clears all sessions and keys and disables SSH for all vsys on the device. The information removed includes:

- Active SSH sessions
- SSH enablement for the current vsys
- PKA keys
- Host keys

Description: Use the **di** commands to configure the NetScreen device to perform DI (Deep Inspection) on packets that use specified protocols.

DI is a mechanism for filtering traffic permitted by the NetScreen firewall. Deep Inspection examines Layer 3 and 4 packet headers and Layer 7 application content and protocol characteristics in an effort to detect and prevent any attacks or anomalous behavior that might be present.

Note: To use this command, you must install an advanced key.

Syntax

get

```
get di
{
  disable_tcp_checksum |
  service
  {
    dns
    {
      report_unexpected |
      report_unknowns |
      udp_message_limit
    } |
    ftp
    {
      line_length |
      password_length |
      pathname_length |
      sitestring_length |
      username_length
    } |
  }
}
```

```
http
{
    content_type_length |
    cookie_length |
    header_length |
    host_length |
    referer_length |
    request_length |
    user_agent_length
} |

imap
{
    flag_length |
    line_length |
    mbox_length |
    pass_length |
    ref_length |
    user_length
} |

pop3
{
    apop_length |
    line_length |
    max_msg_num |
    pass_length |
    user_length
} |

smtp
{
    cmdline_length |
    domain_length |
    num_rcpt |
    path_length |
    replyline_length |
    textline_length |
```

```
        user_length
      }
    }
  }

set

set di
{
  disable_tcp_checksum |
  service
  {
    dns
    {
      report_unexpected number |
      report_unknowns number |
      udp_message_limit number
    } |
    ftp
    {
      line_length number |
      password_length number |
      pathname_length number |
      sitestring_length number |
      username_length number
    } |
    http
    {
      content_type_length number |
      cookie_length number |
      header_length number |
      host_length number |
      referer_length number |
      request_length number |
      user_agent_length number
    }
  }
}
```

```
    } |
imap
{
    flag_length number |
    line_length number |
    mbox_length number |
    pass_length number |
    ref_length number |
    user_length number
} |
pop3
{
    apop_length number |
    line_length number |
    max_msg_num number |
    pass_length number |
    user_length number
} |
smtp
{
    cmdline_length number |
    domain_length number |
    num_rcpt number |
    path_length number |
    replyline_length number |
    textline_length number |
    user_length number
}
}
```

unset

```
unset di
{
  disable_tcp_checksum |
  service
  {
    dns { report_unexpected | report_unknowns | udp_message_limit } |
    ftp
    {
      line_length |
      password_length |
      pathname_length |
      sitestring_length |
      username_length
    } |
    http
    {
      content_type_length |
      cookie_length |
      header_length |
      host_length |
      referer_length |
      request_length |
      user_agent_length
    } |
    imap
    {
      flag_length |
      line_length |
      mbox_length |
      pass_length |
      ref_length |
      user_length
    } |
  }
}
```

```
pop3
{
  apop_length |
  line_length |
  max_msg_num |
  pass_length |
  user_length
} |
smtp
{
  cmdline_length |
  domain_length |
  num_rcpt |
  path_length |
  replyline_length |
  textline_length |
  user_length
}
}
```

Keywords and Variables

disable_tcp_checksum

```
get disable_tcp_checksum
set disable_tcp_checksum
unset disable_tcp_checksum
```

disable_tcp_checksum Disables the TCP checksum operation. The device uses TCP checksums in exchanged packets to detect TCP transmission errors.
Because the checksum operation uses up processor resources, it may be useful to disable it.
The device performs the checksum operation by default.

Examples: The following command disables the checksum operation:

```
set di disable_dhcp_checksum
```

The following command enables the checksum operation:

```
unset di disable_dhcp_checksum
```

dns

```
get di service dns { ... }
set di service dns { ... }
unset di service dns { ... }
```

dns Determines how the NetScreen device evaluates DNS traffic.

- **report_unexpected** *number* Enables or disables reporting of unexpected DNS parameters. A value of 0 disables, 1 enables.
- **report_unknowns** *number* Enables or disables reporting of unknown DNS parameters. A value of 0 disables, 1 enables.
- **udp_message_limit** *number* Specifies the maximum size of a UDP message in bytes. The default is 512 bytes.

ftp

```
get di service ftp { ... }  
set di service ftp { ... }  
unset di service ftp { ... }
```

- ftp** Determines how the NetScreen device evaluates the length of items contained in FTP packets. The device compares the lengths to threshold settings, which define the limits of normal FTP traffic. Any traffic that exceeds such a limit is abnormal, and may contain protocol anomalies. In response, the device generates a log message.
- The range of each setting is 1 byte to 8192 bytes, inclusive.
- **line_length** *number* Specifies the maximum number of bytes for a FTP line. The default setting is 1024 bytes.
 - **password_length** *number* Specifies the maximum number of bytes for a FTP password. The default setting is 64 bytes.
 - **pathname_length** *number* Specifies the maximum number of bytes for a FTP pathname. The default setting is 512 bytes.
 - **sitestring_length** *number* Specifies the maximum number of bytes for a FTP site string. The default setting is 512 bytes.
 - **username_length** *number* Specifies the maximum number of bytes for a FTP username. The default setting is 32 bytes.

http

```
get di service http { ... }  
set di service http { ... }  
unset di service http { ... }
```

http

Determines how the NetScreen device evaluates the length of items contained in HTTP packets. The device compares the lengths to threshold settings, which define the limits of normal HTTP traffic. Any traffic that exceeds such a limit is abnormal, and may contain protocol anomalies. In response, the device generates a log message.

The range of each setting is 1 byte to 8192 bytes, inclusive.

- **content_type_length** *number* Specifies the maximum number of bytes for a HTTP header Content Type field. This field specifies the media type of the data contained in the HTTP packet. The default setting is 64 bytes.
- **cookie_length** *number* Specifies the maximum number of bytes in a cookie. The default setting is 2048 bytes.

Note: Cookies that exceed the cookie length setting can match the protocol anomaly HTTP-HEADER-OVERFLOW and produce unnecessary log records. If the NetScreen device generates too many log records for this anomaly, increase the cookie length setting.

- **header_length** *number* Specifies the maximum number of bytes for an HTTP packet header. The default setting is 1024 bytes.
- **host_length** *number* Specifies the maximum number of bytes for a HTTP header host, which may be an Internet host domain name or IP address. The default setting is 64 bytes.
- **referer_length** *number* Specifies the maximum number of bytes for the header referer field, which the client uses to specify the address URI (Uniform Resource Identifier). The URI is a formatted string that identifies a network resource by a characteristic such as a name or a location. The default setting is 1024 bytes.
- **request_length** *number* Specifies the maximum number of bytes for a HTTP request, which includes information such as a network resource identifier, the method to apply to the resource, and the protocol version. The default setting is 4096 bytes.
- **user_agent_length** *number* Specifies the maximum number of bytes for a HTTP header user-agent field, which contains information about the user agent that originated the request. The default setting is 258 bytes.

imap

```
get di service imap { ... }  
set di service imap { ... }  
unset di service imap { ... }
```

- imap** Controls how the Sensor handles IMAP (Internet Message Access Protocol) packets, by specifying threshold parameters that define the limits and boundaries for normal IMAP traffic. Traffic that exceeds these boundaries is abnormal, and might contain protocol anomalies. The IMAP threshold parameters are as follows:
- **flag_length** *number* Specifies the maximum number of bytes for an IMAP flag. If the Sensor sees an IMAP flag containing more bytes than this maximum, it can consider the packet a possible protocol anomaly. The default setting is 64 bytes.
 - **line_length** *number* Specifies the maximum number of bytes for an IMAP line. If the Sensor sees an IMAP line containing more bytes than this maximum, it can consider the packet a possible protocol anomaly. The default setting is 2048 bytes.
 - **mbox_length** *number* Specifies the maximum number of bytes for an IMAP mailbox. If the Sensor sees an IMAP mailbox containing more bytes than this maximum, it can consider the packet a possible protocol anomaly. The default setting is 64 bytes.
 - **pass_length** *number* Specifies the maximum number of bytes for an IMAP password. If the Sensor sees an IMAP password containing more bytes than the specified maximum, it can consider the packet a possible protocol anomaly. The default setting is 64 bytes.
 - **ref_length** *number* Specifies the maximum number of bytes for an IMAP reference. If the Sensor sees an IMAP reference containing more bytes than this maximum, it generates a protocol anomaly. The default setting is 64 bytes.
 - **user_length** *number* Specifies the maximum number of bytes for an IMAP user name. If the Sensor sees an IMAP user name containing more bytes than this maximum, it can consider the packet a possible protocol anomaly. The default setting is 64 bytes.

pop3

```
get di service pop3 { ... }  
set di service pop3 { ... }  
unset di service pop3 { ... }
```

pop3 Controls how the Sensor handles POP3 (Post-Office Protocol 3) packets by specifying threshold parameters, which define the limits and boundaries for normal POP3 traffic. Traffic that exceeds these boundaries is abnormal, and might contain protocol anomalies. The POP3 threshold parameters are as follows:

- **apop_length** *number* Specifies the maximum number of bytes for an APOP. If the Sensor sees an APOP containing more bytes than the specified maximum, it can consider the packet a possible protocol anomaly. The default is 100 bytes.
- **line_length** *number* Specifies the maximum number of bytes for a POP3 line. If the Sensor sees an POP3 line containing more bytes than the specified maximum, it can consider the packet a possible protocol anomaly. The default is 512 bytes.
- **max_msg_num** *number* Specifies the maximum message number for a POP3 message. If the Sensor sees a POP3 message number that is higher than the specified maximum, it can consider the packet a possible protocol anomaly. The default is 1000000 bytes.
- **pass_length** *number* Specifies the maximum number of bytes in a POP3 password. If the Sensor sees an POP3 password containing more bytes than the specified maximum, it can consider the packet a possible protocol anomaly. The default is 64 bytes.
- **user_length** *number* Specifies the maximum number of bytes in an POP3 user name. If the Sensor sees an POP3 user name containing more bytes than the specified maximum, it can consider the packet a possible protocol anomaly. The default is 64 bytes.

smtp

```
get di service smtp { ... }  
set di service smtp { ... }  
unset di service smtp { ... }
```

- smtp** Use the SMTP (Simple Mail Transfer Protocol) threshold parameters to control how the Sensor handles SMTP packets. The threshold parameters define the boundaries of normal SMTP traffic. Traffic that exceeds these boundaries is considered abnormal, and might contain protocol anomalies. The default SMTP thresholds are below:
- **cmdline_length** *number* Specifies the maximum number of bytes in an SMTP command line positioned before the data. If the Sensor sees an SMTP command line containing more bytes than the specified maximum before the data, it can consider the packet a possible protocol anomaly. The default setting is 1024 bytes.
 - **domain_length** *number* Specifies the maximum number of bytes in an SMTP domain name in the RCPT field of an SMTP message. If the Sensor sees an SMTP domain name in the RCPT field containing more bytes than the specified maximum, it can consider the packet a possible protocol anomaly. The default setting is 64 bytes.
 - **num_rcpt** *number* Specifies the maximum number of recipients for an SMTP message. If the Sensor sees an SMTP message containing more recipients than the specified maximum, it can consider the message a possible protocol anomaly. The default setting is 100 recipients.
 - **path_length** *number* Specifies the maximum number of bytes in an SMTP path in the RCPT field of an SMTP message. If the Sensor sees an SMTP path in the RCPT field containing more bytes than the specified maximum, it can consider the packet a possible protocol anomaly. The default setting is 256 bytes.
 - **replyline_length** *number* Specifies the maximum number of bytes in an SMTP reply line from the server. If the Sensor sees an SMTP reply line from the server containing more bytes than the specified maximum, it can consider the packet a possible protocol anomaly. The default setting is 512 bytes.

- **textline_length** *number* Specifies the maximum number of bytes in an SMTP text line after the data. If the Sensor sees an SMTP text line containing more bytes than the specified maximum after the data, it can consider the packet a possible protocol anomaly. The default setting is 1024 bytes.
- **user_length** *number* Specifies the maximum number of bytes in an SMTP user name. If the Sensor sees an SMTP user name containing more bytes than the specified maximum, it can consider the packet a possible protocol anomaly. The default setting is 256 bytes.

dip

Description: Use **dip** commands to set up a Dynamic IP (DIP) group, display DIP group information, or assign the same IP address from a port-translating DIP pool to a host that originates multiple concurrent sessions ("sticky dip").

A DIP group contains one or more DIP pools, each pool consisting of a range of IP addresses defined on a Layer 3 security zone interface, Layer 3 security zone extended interface, or numbered tunnel interface. When multiple NetScreen devices are in a high availability cluster, a policy requiring source address translation and referencing a DIP pool defined on one virtual security interface (VSI) can result in dropped traffic. When that traffic arrives at a physical NetScreen device on which the DIP pool specified in the policy belongs to a VSI in an inactive virtual security device (VSD), the device drops the traffic because it cannot find the specified DIP pool to use for address translation. If, instead, the policy references a DIP group that contains DIP pools on different egress VSIs, the NetScreen device receiving the traffic can use the DIP pool belonging to the VSI for its active VSD.

Note: *If the range of addresses in a DIP pool is in the same subnet as the interface IP address, the pool must exclude the interface IP address, router IP addresses, and any mapped IP or virtual IP addresses (MIPs and VIPs) that might also be in that subnet. If the range of addresses is in the subnet of an extended interface, the pool must exclude the extended interface IP address.*

Syntax

get

```
get dip [ all ]
```

set

```
set dip
{
  group { id_num1 [ member id_num2 ] } |
  sticky
}
```

unset

```
unset dip { group { id_num1 [ member id_num2 ] } | sticky }
```

Keywords and Variables

group

```
set dip group id_num1 [ member id_num2 ]  
unset dip group id_num1 [ member id_num2 ]
```

group Creates a DIP group or adds a DIP pool to a group. *id_num1* is the identification number you assign to the new DIP group. **member** *id_num2* specifies the identification number of a DIP pool.

Example: The following commands create DIP pools and a DIP group.

- DIP pool with ID 5 for interface ethernet3, which has IP address 1.1.1.1/24
- DIP pool with ID 6 for interface ethernet3:1, which has IP address 1.1.1.2/24
- DIP group with ID number 7. Both DIP pools added to the DIP group

```
set interface ethernet3 dip 5 1.1.1.10 1.1.1.10  
set interface ethernet3:1 dip 6 1.1.1.11 1.1.1.11  
set dip group 7  
set dip group 7 member 5  
set dip group 7 member 6
```

sticky

```
set dip sticky  
unset dip sticky
```

sticky Specifies that the NetScreen device assigns the same IP address to a host for multiple concurrent sessions.

Description: Use **dns** commands to configure Domain Name System (DNS) or to display DNS configuration information.

DNS allows network devices to identify each other using domain names instead of IP addresses. Support for DNS is provided by a DNS server, which keeps a table of domain names with associated IP addresses. For example, using DNS makes it possible to reference locations by domain name (such as www.juniper.net) in addition to using the routable IP address (which for www.juniper.net is 209.125.148.135).

DNS translation is supported in all the following applications:

- Address Book
- Syslog
- E-mail
- WebTrends
- Websense
- LDAP
- SecurID
- RADIUS
- NetScreen-Global PRO

Before you can use DNS for domain name/address resolution, you must enter the addresses for DNS servers (the primary and secondary DNS servers) in the NetScreen device.

Syntax

clear

```
clear [ cluster ] dns
```

exec

```
exec dns refresh
```

get

```
get dns
{
  host { cache | report | settings } |
  name dom_name
}
```

set

```
set dns host
{
  dns1 ip_addr |
  dns2 ip_addr |
  schedule time [ interval number ]
}
```

unset

```
unset dns host { dns1 | dns2 | schedule }
```

Keywords and Variables

cluster

```
clear [ cluster ] dns
```

cluster Propagates the **clear** operation to all other devices in a NSRP cluster.

host

```
get dns host { ... }  
set dns host { ... }  
unset dns host { ... }
```

- host**
- **cache** Displays the DNS cache table.
 - **dns1** *ip_addr* Specifies the primary DNS server.
 - **dns2** *ip_addr* Specifies the backup DNS server.
 - **name** The domain name of the host, listed in the DNS table.
 - Using the **name** option with **get** directs the NetScreen device to look up an IP address for the given domain name.
 - Using the **name** option with **set** places an entry in the DNS table, representing a host device with a host name and IP address. This allows you to reach the host from the NetScreen device using the host name. For example, executing **set dns host name acme 2.2.2.25** creates a DNS table entry for a host at address **2.2.2.25**, with a host name of **acme**. This allows you to reach the host from the NetScreen device, as with the command **ping acme**.
- Note:** The DNS table is local to the NetScreen device, and functions only as a proxy for the actual DNS server. Consequently, other network nodes cannot query the listed names using the NetScreen device. The main purpose of the table is to let you create an alias for an external host and to access that host from the NetScreen device.
- **report** Displays the DNS lookup table.

- **schedule** *time* Specifies the time of day to refresh DNS entries. The format of this parameter is hh:mm. The **interval** *number* parameter specifies a 4-, 6-, 8-, or 12-hour interval between DNS table refresh operations. The default interval is 24 hours; that is, once a day at the scheduled DNS lookup time. Use this option to refresh the DNS table more frequently.
- **settings** Displays DNS settings, including IP addresses, refresh setting, and the number of UDP sessions.

Examples: The following command sets up a host as the primary DNS server at IP address **1.2.2.45**:

```
set dns host dns1 1.2.2.45
```

The following command schedules a refresh time at **23:59** each day, and a DNS table refresh interval of 12 hours:

```
set dns host schedule 23:59 interval 12
```

refresh

```
exec dns refresh
```

refresh	Refreshes all DNS entries. Using the option directs the NetScreen device to perform a manual DNS lookup.
----------------	--

domain

Description: Use the **domain** commands to set or display the domain name of the NetScreen device.

A *domain name* is a character string that identifies the NetScreen device. This name allows other devices to access the NetScreen device through a DNS server, thus identifying the device without using an explicit IP address.

Syntax

get

```
get domain
```

set

```
set domain name_str
```

unset

```
unset domain
```

Keywords and Variables

Variable Parameter

name_str Defines the domain name of the NetScreen device.

Example: The following command sets the domain of the NetScreen device to **acme**:

```
set domain acme
```

downgrade

Description: Use the **downgrade** command to downgrade the ScreenOS firmware from ScreenOS 5.0.X to ScreenOS 4.0.X.

To use this command to perform a downgrade, you must have the following.

- Root or Read-Write privileges to the NetScreen device
- A console connection to the NetScreen device
- A TFTP server application running on your computer
- An Ethernet connection from your computer to the NetScreen device (to transfer data from the TFTP server on your computer)
- A ScreenOS 4.0.X image file saved to the TFTP server folder on your computer
- A configuration file that was saved in ScreenOS 4.0.X (configurations saved in ScreenOS 5.0.0 are not supported by ScreenOS 4.0.X)

For information on the downgrade process, refer to the *NetScreen ScreenOS Migration Guide* for ScreenOS 5.0.0.

Warning: *Execute this command with extreme caution. Before execution, refer to the NetScreen ScreenOS Migration Guide for further details.*

Syntax

exec

exec downgrade

Keywords and Variables

None.

Downgrades and NSM

Before downgrading ScreenOS from 5.x to 4.x on a device that uses NSM (Juniper Networks NetScreen-Security Manager), execute the following commands:

```
unset nsmgmt enable
unset nsmgmt init otp
unset nsmgmt init id
unset nsmgmt server primary
del nsmgmt keys
save
```

envvar

Description: Use the **envvar** commands to define environment variables.

The NetScreen device uses environment variables to make special configurations at startup.

Syntax

get

```
get envvar [ resource ]
```

set

```
set envvar string
```

unset

```
unset envvar string
```

Keywords and Variables

Variable Parameter

```
set envvar string  
unset envvar string
```

string The location of the environment variables files.

Example: The following command defines the location of the system configuration as **file2.cfg** in **slot2**:

```
set envvar config=slot2:file2.cfg
```


resource

get envvar resource

resource

Displays the following information:

- (max-session) Maximum number of sessions
- (max-sa) Maximum number of security associations (SAs)
- (max-l2tp-tunnel) Maximum number of L2TP tunnels

Description: Use the **event** commands to display or clear event log messages.

The *event log* monitors and records system events and network traffic. The NetScreen device categorizes logged system events by the following severity levels:

- **Alert:** Messages for multiple user authentication failures and other firewall attacks not included in the emergency category.
- **Critical:** Messages for URL blocks, traffic alarms, high availability (HA) status changes, and global communications.
- **Debugging:** All messages.
- **Emergency:** Messages concerning SYN attacks, Tear Drop attacks, and Ping of Death attacks.
- **Error:** Messages for admin log on failures.
- **Information:** Any kind of message not specified in other categories.
- **Notification:** Messages concerning link status changes, traffic logs, and configuration changes.
- **Warning:** Messages for admin logins and logouts, failures to log on and log out, and user authentication failures, successes, and timeouts.

The event log displays the date, time, level and description of each system event.

Syntax

clear

```
clear [ cluster ] event [ end-time time ]
```

get

```
get event  
  [ module name_str ]  
    [ level  
      {  
        alert |  
        critical |  
        debug |  
        emergency |  
        error |  
        information |  
        notification |  
        warning  
      }  
    ]  
    [ type id_num1 ] [ -id_num2 ]  
      [ start-date date [ time ] ] [ end-date date [ time ] ]  
        [ start-time time ] [ end-time time ]  
          [ include string ] [ exclude string ]  
            [ src-ip ip_addr1 [ -ip_addr2 | src_netmask mask ] ]  
              [ dst-ip ip_addr1 [ -ip_addr2 | dst_netmask mask ] ]  
  
sort-by  
  {  
    date  
      [ start-date date [ time ] ]
```

```
        [ end-date date [ time ] ]
dst-ip [ ip_addr [ -ip_addr | dst-netmask mask ] ]
src-ip [ ip_addr [ -ip_addr | src-netmask mask ] ]
time
    [ start-time time ]
    [ end-time time ]
}
]
```

Keywords and Variables

cluster

clear cluster event [...]

cluster Propagates the **clear** operation to all other devices in a NSRP cluster.

dst-ip

get event dst-ip *ip_addr* [*-ip_addr* | **dst-netmask** *mask*]
get event sort by dst-ip [*ip_addr* [*-ip_addr* | **dst-netmask** *mask*]]

dst-ip Directs the device to display event logs with the specified destination IP address or address range. The device can also sort event logs by destination IP address.

include | exclude

get event [...] [**include** *string*] [**exclude** *string*] [...]

include Directs the device to exclude or include events containing a specifies string of characters (*string*).
exclude

level

```
get event module name_str level { ... }
```

- level** Specifies the priority level of the event message. The priority levels are as follows:
- **emergency** (Level 0) The system is unusable.
 - **alert** (Level 1) Immediate action is necessary.
 - **critical** (Level 2) The event affects functionality.
 - **error** (Level 3) Error condition exists.
 - **warning** (Level 4) The event might affect functionality.
 - **notification** (Level 5) The event is a normal occurrence.
 - **information** (Level 6) The event generates general information about normal operation.
 - **debug** (Level 7) The event generates detailed information for troubleshooting purposes.

module

```
get event module name_str [ ... ]
```

- module** Specifies the name of the system module that generated the event.

src-ip

```
get event src-ip ip_addr1 [ -ip_addr2 | src-netmask mask ]  
get event sort by src-ip ip_addr1 [ -ip_addr2 | src-netmask mask ]
```

- src-ip** Directs the device to sort event logs by source IP address. The device can also display event logs with the specified source IP address or address range.

start-time | end-time

```
clear [ cluster ] event end-time time
get event [ ... ] [ start-time time ] [ end-time time ] [ ... ]
```

end-time time Specifies the lower and upper ends of a range of times for an event. When you specify a start-time and/or end-time, the device sorts or filters the event logs based on the specified times, regardless of the date. The format is: **hh:mm:ss**.

start-time time When you use the **end-time** option with the **clear event** command, you specify the date and optionally the time in the following format: **mm/dd/yy-hh:mm:ss**.

Example: The following command clears all events generated before May 1, 2002 at 11:30am:

```
get event end-time 05/01/02-11:30:00
```

start-date | end-date

```
get event [ start-date date [ time ] ] [ end-date date [ time ] ]
get event sort-by date [ start-date date [ time ] ] [ end-date date [ time ] ]
```

start-date end-date Specifies the lower and upper ends of a range of times for an event. The format is: **mm/dd/yy-hh:mm:ss**

You can omit the year (the current year is the default), or express the year using the last two digits or all four digits. The hour, minute, and second are optional. The delimiter between the date and the time can be a dash or an underscore:

12/31/2001-23:59:00
12/31/2001_23:59:00

type

```
get event module name_str level { ... } type id_num1 [ ... ]
```

type Specifies a priority level or a range of priority levels.

exit

Description: Use the **exit** command to exit a command context, a virtual system, or to terminate and log out from a CLI session.

Syntax

exit

Keywords and Variables

None.

Example: The following **exit** command exits the context of policy ID 1 and returns the command context to the top command level:

```
ns-> set policy id 1
ns(policy:1)-> set dst-addr 2.2.2.5/32
ns(policy:1)-> exit
ns->
```

Notes

When issuing the **exit** command at the top command level (that is, not from within a command context), you must log back in to the console to configure a NetScreen device.

failover

Description: Use the **failover** commands to configure failover settings on the NetScreen device.

Syntax

set

```
set failover
{
  auto |
  holddown number |
  type { track-ip | tunnel-if }
}
```

unset

```
unset failover
{
  auto |
  holddown
}
```

exec

```
exec failover
{
  force |
  revert
}
```


Keywords and Variables

auto

```
set failover auto
unset failover auto
```

auto Directs the NetScreen device to automatically fail over from the primary interface to the backup and from the backup interface to the primary. By default, failover is manual (the administrator must use the CLI or WebUI to switch from the primary interface to the backup and from the backup interface to the primary).

force

```
exec failover force
```

force Forces traffic to be switched to the backup interface.

holddown

```
set failover holddown number
unset failover holddown
```

holddown Specifies the time interval (*number*), in seconds, the NetScreen device delays failover actions. This value has an effect in the following situations:

- The NetScreen device switches traffic to the backup interface.
- The NetScreen device switches traffic from the backup interface to the primary interface, when the primary interface becomes available again.

The default holddown interval 30 seconds.

Example: The following command sets a failover delay of 45 seconds:

```
set failover holddown 45
```

revert

exec failover revert

revert Forces traffic to be switched from the backup interface to the primary.

type

set failover type { track-ip | tunnel-if }

type Specifies the type of event that determines interface failover. You can specify the following types:

- **track-ip** instructs ScreenOS to use IP tracking to determine failover.
- **tunnel-if** instructs ScreenOS to use VPN tunnel status to determine failover.

Description: Use the **file** commands to clear or display information for files stored in the flash memory.

Syntax

clear

```
clear [ cluster ] file dev_name:filename
```

get

```
get file [ filename | info ]
```

Keywords and Variables

Variable Parameters

```
clear [ ... ] file dev_name:filename
```

```
get file filename
```

dev_name:filename Deletes the file with the name *filename* from the flash card memory.

filename Defines the file name stored in the flash card memory.

Examples: The following command deletes a file named **myconfig** in the flash memory on the memory board:

```
clear file flash:myconfig
```

The following command displays information for the file named **corpnet** from the flash card memory:

```
get file corpnet
```

cluster

clear cluster file *dev_name:filename*

cluster Propagates the **clear** operation to all other devices in a NSRP cluster.

info

get file info

info Displays the base sector and address.

firewall

Description: Use the **firewall** commands to enable or disable logging of dropped packets targeting an interface address on the NetScreen device.

Note: Juniper Networks NetScreen devices perform most firewall services at the security zone level. You configure individual zones to perform these services. For more information, see the ["zone" on page 551](#).

Syntax

get

```
get firewall
```

set

```
set firewall log-self  
    [ exclude ]  
        [ icmp | ike | multicast | snmp ]
```

unset

```
unset firewall log-self  
    [ exclude ]  
        [ icmp | ike | multicast | snmp ]
```

Keywords and Variables

firewall

get firewall

firewall Displays the settings for logging dropped ICMP, IKE, multicast, and SNMP packets destined for the NetScreen device. Log entries appear in the self log.

log-self

```
set firewall log-self [ exclude ] [ icmp | ike | multicast | snmp ]  
unset firewall log-self [ exclude ] [ icmp | ike | multicast | snmp ]
```

log-self Directs the NetScreen device to log or not log dropped packets and pings in the self log. Using the **exclude** switch directs the device not to perform logging at all or for specified traffic types.

- **icmp** Enables or disables logging of ICMP (Internet Control Message Protocol) packets.
- **ike** Enables or disables logging of dropped IKE (Internet Key Exchange) packets.
- **multicast** Enables or disables logging of multicast packets.
- **snmp** Enables or disables logging of dropped Simple Network Management Protocol (SNMP) packets.

Entering the **set firewall log-self** command without any other keywords enables logging to the self log. (By default, logging to the self log is enabled.) Entering the **unset firewall log-self** command without any other keywords disables it.

flow

Description: Use the **flow** commands to determine how the NetScreen device manages packet flow.

The device can regulate packet flow in the following ways:

- Enable or disable DNS replies when there is no matching DNS request.
- Pass or block packets containing destination MAC addresses that are not in the MAC learning table.
- Set or display the initial session timeout values.
- Control or prevent packet fragmentation.

Syntax

get

```
get flow [ perf | tcpmss ]
```

set

```
set flow
{
  aging { early-ageout number | high-watermark number | low-watermark number }
  all-tcp-mss [ number ] |
  allow-dns-reply |
  gre-in-tcp-mss |
  gre-out-tcp-mss |
  hub-n-spoke-mip |
  initial-timeout number |
  mac-flooding |
  max-frag-pkt-size number |
  no-tcp-seq-check |
  path-mtu |
  tcp-mss [ number ] |
```

```
tcp-rst-invalid-session |  
tcp-syn-check |  
tcp-syn-check-in-tunnel  
}
```

unset

unset flow

```
{  
aging { early-ageout | high-watermark | low-watermark }  
allow-dns-reply |  
gre-in-tcp-mss |  
gre-out-tcp-mss |  
hub-n-spoke-mip |  
initial-timeout |  
mac-flooding |  
max-frag-pkt-size |  
no-tcp-seq-check |  
path-mtu |  
tcp-mss |  
tcp-rst-invalid-session |  
tcp-syn-check |  
tcp-syn-check-in-tunnel  
}
```


Keywords and Variables

aging

```
set flow aging early-ageout number
set flow aging { high-watermark number | low-watermark number }
unset flow aging { early-ageout | high-watermark | low-watermark }
```

aging

Directs the NetScreen device to begin aggressively aging out sessions when the number of entries in the session table exceeds the high-watermark setting, and then stop when the number of sessions falls below the low-watermark setting. When the session table is in any other state, the normal session timeout value is applied—for TCP, session timeout is 30 minutes; for HTTP, it is 5 minutes; and for UDP, it is 1 minute. During the time when the aggressive aging out process is in effect, the NetScreen device ages out sessions—beginning with the oldest sessions first—at the rate you specify.

- **early-ageout** *number* Defines the the ageout value before the NetScreen device aggressively ages out a session from its session table. The value you enter can be from 2 to 10 units, each unit representing a 10-second interval. The default early-ageout value is 2, or 20 seconds.
- **high-watermark** *number* Sets the point at which the aggressive aging out process begins. The number you enter can be from 1 to 100 and indicates a percent of the session table capacity in 1% units. The default is 100, or 100%.
- **low-watermark** *number* Sets the point at which the aggressive aging out process ends. The number you enter can be from 1 to 10 and indicates a percent of the session table capacity in 10% units. The default is 10, or 100%.

Example: The following commands activate the aggressive aging-out process when the session table reaches 70% capacity and deactivate the process when it drops below 60%, and set the aggressive ageout value at 30 seconds:

```
set flow aging low-watermark 60
set flow aging high-watermark 70
set flow aging early-ageout 3
```

allow-dns-reply

```
set flow allow-dns-reply
unset flow allow-dns-reply
```

allow-dns-reply Allows an incoming DNS reply packet without a matched request.

If **allow-dns-reply** is disabled and an incoming UDP first-packet has dst-port 53, the device checks the DNS message packet header to verify that the QR bit is 0 (which denotes a query message). If the QR bit is 1 (which denotes a response message) the device drops the packet, does not create a session, and increments the 'illegal pak' flow counter for the interface.

By default, **allow-dns-reply** is disabled. Enabling **allow-dns-reply** directs the device to skip the check.

all-tcp-mss

```
set flow all-tcp-mss number
unset flow all-tcp-mss
```

all-tcp-mss Sets the TCP-MSS (TCP-Maximum Segment Size) value for all TCP SYN packets with an MSS value greater than that specified by the **tcp-mss** option (described below). The device resets the value whether the packets are clear-text or encrypted. By contrast, use the **tcp-mss** option to modify the MSS value only when it is necessary to avoid fragmentation caused by an IPsec operation.

gre-in-tcp-mss

```
set flow gre-in-tcp-mss
unset flow gre-in-tcp-mss
```

gre-in-tcp-mss Specifies inbound GRE TCP-MSS option (64-1420).

gre-out-tcp-mss

```
set flow gre-out-tcp-mss
unset flow gre-out-tcp-mss
```

gre-out-tcp-mss Specifies outbound GRE TCP MSS option (64-1420).

hub-n-spoke-mip

```
set flow hub-n-spoke-mip
unset flow hub-n-spoke-mip
```

hub-n-spoke-mip Permits the NetScreen device to use hub-and-spoke policies for MIP traffic, when the traffic loops back on the same physical interface. This option only has an effect when the interface is bound to the Untrust zone.

initial-timeout

```
set flow initial-timeout number
unset flow initial-timeout
```

initial-timeout Defines the length of time in seconds (*number*, expressed in 10-second intervals) that the NetScreen device keeps an initial session in the session table before dropping it, or until the device receives a FIN or RST packet. The range of time is from 20 seconds (setting of 2) to 300 seconds (setting of 30).

Example: The following command sets the initial timeout value to one minute (60 seconds):

```
set flow initial-timeout 6
```

mac-flooding

```
set flow mac-flooding
unset flow mac-flooding
```

mac-flooding Enables the NetScreen device to pass a packet across the firewall even if its destination MAC address is not in the MAC learning table.

max-frag-pkt-size

```
set flow max-frag-pkt-size number
unset flow max-frag-pkt-size
```

max-frag-pkt-size The maximum allowable size for a packet fragment generated by the NetScreen device. You can set the *number* value between 1024 and 1500 inclusive.
For example, if a received packet is 1540 bytes and **max-frag-pkt-size** is 1460 bytes, the device generates two fragment packets. The first is 1460 bytes and the second is 80 bytes. If you reset **max-frag-pkt-size** to 1024, the first fragment packet is 1024 bytes and the second is 516 bytes.

Example: The following command sets the maximum size of a packet generated by the NetScreen device to 1024 bytes:

```
set flow max-frag-pkt-size 1024
```

no-tcp-seq-check

```
set flow no-tcp-seq-check
unset flow no-tcp-seq-check
```

no-tcp-seq-check Skips the sequence number check in stateful inspection.

path-mtu

```
set flow path-mtu
unset flow path-mtu
```

path-mtu Enables path-MTU (maximum transmission unit) discovery. If the NetScreen device receives a packet that must be fragmented, it sends an ICMP packet suggesting a smaller packet size.

perf

```
get flow perf
```

perf Displays the perf information.

tcp-mss

```
tcp-rst-invalid-session get flow tcpmss
set flow tcpmss [ number ]
unset flow tcpmss
```

tcp-mss Enables the TCP-MSS (TCP-Maximum Segment Size) option. The NetScreen device modifies the MSS value in the TCP packet to avoid fragmentation caused by the IPsec operation.

tcp-rst-invalid-session

```
set flow tcp-rst-invalid-session
unset flow tcp-rst-invalid-session
```

tcp-rst-invalid-session Directs the device to immediately terminate and reset a session when TCP packets have enabled reset bits.

tcp-syn-check

```
set flow tcp-syn-check  
unset flow tcp-syn-check
```

tcp-syn-check Checks the TCP SYN bit before creating a session.

tcp-syn-check-in-tunnel

```
set flow tcp-syn-check-in-tunnel  
unset flow tcp-syn-check-in-tunnel
```

tcp-syn-check-in-tunnel Checks the TCP SYN bit before creating a session for tunneled packets.

Defaults

The default initial timeout value is 2 (20 seconds).

The MAC-flooding feature is enabled by default.

gate

Description: Use the **gate** command to check the number of gates on the NetScreen device, how many are in use, and how many are still available.

Gates are logical access points in the firewall for FTP and similar applications. The NetScreen device creates the gates, then converts a gate for each new session when data traffic occurs.

Syntax

get

get gate

Keywords and Variables

None.

Defaults

The default number of gates on NetScreen devices are:

NetScreen-5000 Series	8192
NetScreen-500	4096
NetScreen-200 Series	1024
NetScreen-100	1024
NetScreen-25/50	256
NetScreen-5XT	256
NetScreen-5GT	256

group

Description: Use the **group** commands to group several addresses or several services under a single name.

A *group* allows you to reference a group of addresses or services by a single name in a policy. This eliminates the need for a separate policy for each address or service. For example, you can create a service group that includes FTP, HTTP, and HTTPS services, and then reference that group in a policy.

Note: Although a single policy might reference a service group with three members, the NetScreen device generates multiple internal rules from that policy. Overusing address and service groups with high member counts can unexpectedly consume internal resources.

Syntax

get

```
get group { address zone [ grp_name ] | service [ grp_name ] }
```

set

```
set group
{
  address zone grp_name [ add name_str ] [ comment string ] |
  service grp_name [ add name_str ] [ comment string ]
}
```

unset

```
unset group
{
  address zone grp_name [ remove mbr_name | clear ] |
  service grp_name [ remove mbr_name | clear ]
}
```


Keywords and Variables

add

```
set group address zone grp_name [ add mbr_name ] [ comment string ]
set group service grp_name [ add mbr_name [ comment string ] ]
```

add *name_str* Adds an address or service named *mbr_name*.

Examples: The following command creates an address group named **engineering** for the Trust zone and adds the address **hw-eng** to the group:

```
set group address trust engineering add hw-eng
```

The following command creates a service group named **inside-sales** and adds the service AOL to the group:

```
set group service inside-sales add AOL
```

address

```
get group address zone [ ... ]
set group address zone grp_name [ ... ]
unset group address zone grp_name [ ... ]
```

address Performs the operation on an address group. The *zone* value specifies the zone to which the address group is bound. This zone is either a default security zone or a user-defined zone. For more information on zones, see ["Zone Names" on page B-1](#).

Example: The following command creates an empty address group (named **headquarters**) for the Trust zone:

```
set group address trust headquarters
```

clear

```
unset group address zone grp_name clear
unset group service grp_name clear
```

clear Removes all the members of an address or service group.

Example: The following command removes all members from an address group (**engineering**) bound to the Trust zone:

```
unset group address trust engineering clear
```

comment

```
set group address zone grp_name [ ... ] [ comment string ]
set group service grp_name [ ... ] [ comment string ]
```

comment Adds a comment *string* to the service group or address group entry.

Example: The following command creates an address group named **engineering** for the Trust zone, adds the address **hw-eng** to the group, and includes a comment about the group:

```
set group address trust engineering add hw-eng comment "Engineering Group"
```

remove

```
unset group address zone grp_name remove name_str
unset group service grp_name remove name_str
```

remove Removes the address (or service) named *name_str*. If you do not specify an address (or service) group member, the **unset group { address | service }** command deletes the entire address group or service group.

Example: The following command removes the address **admin-pc** from the **engineering** address group:

```
unset group address trust engineering remove admin-pc
```

service

```
get group service grp_name
set group service grp_name [ ... ]
unset group service grp_name [ ... ]
```

service *grp_name* Performs the operation on a service group.

Example: The following command creates an empty service group and names it **web_browsing**:

```
set group service web_browsing
```

Notes

Each address group and service group you create must have a unique name. You cannot use the same address group name as a service group name.

You cannot add the predefined address or service named "any" to a group.

While a policy references a group, you cannot remove the group, although you can modify it.

From the console, you can add only one member to a group at a time.

group-expression

Description: Use the **group-expression** commands to set up or display group expressions for use in security policies.

A *group expression* allows you to include or exclude users or user groups, according to NOT, AND, or OR operators. Such expressions are only usable for external users and user groups.

Syntax

get

```
get group-expression
{
  name_str |
  all |
  id number
}
```

set

```
set group-expression name_str
{
  not name_str |
  name_str { and | or } name_str |
  id number |
}
```

unset

```
unset group-expression
{ name_str | id number }
```

Keywords and Variables

Variable Parameters

```
get group-expression name_str  
set group-expression name_str  
unset group-expression name_str
```

name_str The name of the group expression.

all

```
get group-expression all
```

all Specifies all group expressions.

and | or

```
set group-expression name_str name_str and name_str  
set group-expression name_str name_str or name_str
```

and | or Specifies AND or OR relationship between users, user groups, or group expressions.

Example: The following commands create group expressions **SalesM** and **SM_Group**, place them in an OR relationship, and then place **SM_Group** and **Office_1** in an AND relationship:

```
set user-group Sales_Group location external  
set user-group Marketing_Group location external  
set group-expression SalesM Sales_Group or Marketing_Group  
set group-expression SM_Group Office_1 and SalesM
```

id

```
get group-expression id number  
set group-expression name_str id number  
unset group-expression id number
```

id *number* Specifies an identification number for the group expression.

not

```
set group-expression name_str not name_str
```

not Specifies negation.

Example: The following command creates a NOT group expression that does not allow the **Office_1** user:

```
set group-expression Total_Users not Office_1
```

hostname

Description: Use the **hostname** commands to define the NetScreen device name. This name always appears in the console command prompt.

The host name is a character string that identifies the NetScreen device. If you define a host name for the device (such as ns500gate) and a domain name for the device (such as "netscreen," using the [domain](#) command), you can use the host name and domain name (ns500gate.netscreen) as a gateway for a VPN tunnel.

Syntax

get

```
get hostname
```

set

```
set hostname string
```

unset

```
unset hostname
```

Keywords and Variables

Variable Parameters

string Sets the name of the NetScreen device.

Example: The following command changes the NetScreen device hostname to acme:

```
set hostname acme
```

ike

Definition: Use the **ike** commands to define the Phase 1 and Phase 2 proposals and the gateway for an AutoKey IKE (Internet Key Exchange) VPN tunnel, and to specify other IKE parameters.

To establish an AutoKey IKE IPSec tunnel between peer devices, two phases of negotiation are required:

- In Phase 1, the peer devices establish a secure channel in which to negotiate the IPSec SAs.
- In Phase 2, the peer devices negotiate the IPSec SAs for encrypting and authenticating the ensuing exchanges of user data.

The gateway definition identifies the devices or remote users with which the NetScreen device establishes the VPN tunnel.

Syntax

exec

```
exec ike preshare-gen name_str usr_str
```

get

```
get ike
{
  accept-all-proposal |
  ca-and-type |
  cert |
  conn-entry |
  cookies |
  gateway [ name_str ] |
  heartbeat |
  id-mode |
  initial-contact [ all-peers | single-gateway [ name_str ] ] |
  initiator-set-commit |
  member-sa-hold-time |
}
```



```
p1-max-dialgrp-sessions |
p1-proposal name_str |
p1-sec-level |
p2-proposal name_str |
p2-sec-level |
policy-checking |
respond-bad-spi |
responder-set-commit |
soft-lifetime-buffer
}
```

set

Phase 1 Proposal

```
set ike p1-proposal name_str
  [ dsa-sig | rsa-sig | preshare ]
  [ group1 | group2 | group5 ]
  { esp
    { 3des | des | aes128 | aes192 | aes256
      { md5 | sha-1
        [
          days number |
          hours number |
          minutes number |
          seconds number
        ]
      }
    }
  }
```

Phase 2 Proposal

```
set ike p2-proposal name_str
  [ group1 | group2 | group5 | no-pfs ]
  {
    esp [ 3des | des | aes128 | aes196 | aes256 | null ] |
    ah
  }
  [ md5 | null | sha-1
    [
      days number |
      hours number |
      minutes number |
      seconds number ]
    [ kbyte number ]
  ]
}
```

Gateway Tunnel

```
set ike gateway name_str
{
  address { ip_addr | hostname[.dom_name] [ id ] }
  dialup { usr_str | grp_name } |
  dynamic
  {
    string |
    asn1-dn { [ container string ] [ wildcard string ] |
    fqdn string |
    ip-addr string |
    u-fqdn string
  } |
  [ aggressive | main ] [ local-id id_str ]
  [ outgoing-interface interface
```

```
        [ outgoing-zone zone ]
    ]
        [ preshare key_str | seed-preshare key_str ]
        {
            sec-level { basic | compatible | standard } |
            proposal name_str1
                [ name_str2 ] [ name_str3 ] [ name_str4 ]
        }
```

IKE Heartbeat

```
set ike gateway name_str heartbeat
{
    hello number |
    threshold number |
    reconnect number
}
```

Certificates

```
set ike gateway name_str cert
{
    my-cert id_num |
    peer-ca [ id_num | all ] |
    peer-cert-type { pkcs7 | x509-sig }
}
```

NAT-Traversal

```
set ike gateway name_str nat-traversal
[
    udp-checksum |
    keepalive-frequency number
]
```

XAuth

```
set ike gateway name_str xauth
[
  client { any | chap | securid } username name_str password name_str |
  server name_str
  [ chap ] [ query-config ] [ user name_str | user-group name_str ] |
  bypass-auth
]
```

Other IKE Command Switches

```
set ike
{
  accept-all-proposal |
  id-mode { ip | subnet } |
  initial-contact
  [
    all-peers |
    single-gateway name_str
  ] |
  initiator-set-commit |
  member-sa-hold-time number |
  pl-max-dialgrp-sessions { count number | percentage number } |
  policy-checking |
  respond-bad-spi spi_num |
  responder-set-commit |
  single-ike-tunnel name_str |
  soft-lifetime-buffer number
}
```

unset

```
unset ike
{
  accept-all-proposal |
  gateway name_str
  [
    heartbeat { hello | reconnect | threshold } |
    my-cert |
    nat-traversal [ udp-checksum ] |
    peer-ca |
    peer-cert-type |
    xauth
  ] |
  heartbeat { hello | reconnect | threshold } |
  initial-contact |
  initiator-set-commit |
  member-hold-sa |
  p1-max-dialgrp-sessions |
  p1-proposal name_str |
  p2-proposal name_str |
  policy-checking |
  respond-bad-spi |
  responder-set-commit |
  single-ike-tunnel name_str
}
```

Keywords and Variables

accept-all-proposal

```
get ike accept-all-proposal
set ike accept-all-proposal
unset ike accept-all-proposal
```

accept-all-proposal Directs the NetScreen device to accept all incoming proposals. By default, the device accepts only those proposals matching predefined or user-defined proposals. This command is primarily useful when troubleshooting AutoKey IKE tunnels.

address

```
set ike gateway address { ip_addr | name_str } { ... }
```

address Defines the remote IKE gateway address either as an IP address, or as a hostname, or a fully-qualified domain name (FQDN, which is a hostname + domain name). Use this option to set up a site-to-site VPN. **Note:** If you specify a hostname or FQDN that the NetScreen device cannot resolve to an IP address, the IKE gateway is classified as disabled.

Example: The following command specifies www.juniper.net as the address of a remote IKE gateway named ns1, define the preshared key as 7a850wq, and specify the Phase 1 security level as compatible¹:

```
set ike gateway ns1 address www.juniper.net preshare 7a850wq sec-level compatible
```

1. The *compatible* security level for Phase 1 negotiations includes the following four proposals: pre-g2-3des-sha, pre-g2-3des-md5, pre-g2-des-sha, and pre-g2-des-md5.

aggressive | main

```
set ike gateway name_str { ... } aggressive [ ... ]  
set ike gateway name_str { ... } main [ ... ]
```

aggressive | main

Defines the mode used for Phase 1 negotiations. Use Aggressive mode only when you need to initiate an IKE key exchange without ID protection, as when a peer unit has a dynamically assigned IP address. Main mode is the recommended key-exchange method because it conceals the identities of the parties during the key exchange.

ca-and-type

```
get ike ca-and-type
```

ca-and-type

Displays the supported certificate authorities (CAs) and certificate types.

cert

```
get ike cert  
set ike gateway name_str cert my-cert id_num  
set ike gateway name_str cert peer-ca [ id_num | all ]  
set ike gateway name_str cert peer-cert-type { pkcs7 | 509-sig
```

cert

Uses a digital certificate to authenticate the VPN initiator and recipient.

gateway *name_str* cert

Specifies which certificates to use.

- **my-cert *name_str*** Specifies a particular certificate when the local NetScreen device has multiple loaded certificates.
- **peer-ca *name_str*** Specifies a preferred CA (certificate authority).
- **peer-cert-type { pkcs7 | x509 }** Specifies a preferred type of certificate (PKCS7 or X509).
If you set the **peer-ca** and **peer-cert-type** values, the device inserts them in any certificate request it sends to the peer. If the peer has multiple local certificates, these values help the peer select a certificate.

Note: The NetScreen device does *not* use the **peer-ca** or **peer-cert-type** settings to check certificates received from the peer.

If possible, the peer should send a certificate issued by the **peer-ca** CA. However, if the peer sends a certificate issued by a different CA, the NetScreen device searches local memory for the certificate of the issuing CA; if the search is successful, the device accepts the peer certificate. If the search is unsuccessful, the device uses a certificate issued by a different CA.

conn-entry

get ike conn-entry

conn-entry Displays the Connection Entry Table.

cookies

get ike cookies

cookies Displays the cookie table, and the total number of dead and active cookies.

dialup

set ike gateway *name_str* **dialup** { *usr_str* | *grp_name* } [...]

dialup Identifies an IKE dialup user (*usr_str*) or dialup group (*grp_name*). Use this option to set up a dialup VPN. To specify a user's attributes, use the **set user** command. (To specify dialup group attributes, use the **set user-group** command.)

dynamic

```
set ike gateway name_str dynamic { ... } [ ... ]
```

dynamic

Specifies the identifier for the remote gateway with a dynamic IP address. Use this option to set up a VPN with a gateway that has an unspecified IP address.

- *string* A string you can use as a peer ID.
- **asn1-dn** [**container**] [**wildcard**] *string* The ASN1 domain name. The **container** switch treats *string* as a container. The **wildcard** switch treats *string* as a wild card.
- **fqdn** The fully-qualified domain name (such as www.acme.com).
- **ip_addr** *string* The IP address of the remote gateway interface.
- **u-fqdn** *string* The user fully-qualified domain name (such as admin@acme.com).

gateway

```
get ike gateway  
set ike gateway name_str { ... } [ ... ]  
unset ike gateway { ... }
```

gateway

Configures or displays settings for a remote tunnel gateway.

heartbeat

```
get ike heartbeat  
set ike gateway name_str heartbeat { ... }  
unset ike gateway heartbeat { ... }
```

heartbeat

Specifies the IKE heartbeat protocol parameters.

- **hello** *number* Sets the IKE heartbeat protocol interval (in seconds).
- **reconnect** *number* Sets the quiet interval (in seconds) that elapses before the NetScreen device reconnects a failed tunnel.
- **threshold** *number* Sets the number of retries before the NetScreen device considers the connection lost and removes all Phase 1 and Phase 2 keys related to this gateway.

id-mode

```
get ike id-mode
set ike id-mode ip
set ike id-mode subnet
```

id-mode Defines the IKE ID mode in the Phase 2 exchange as either a host (IP) address or a gateway (subnet). If you use the **ip** switch, the device sends no Phase 2 ID. If you choose the **subnet** switch, the device sends proxy Phase 2 IDs. (Use the **ip** switch when setting up a VPN tunnel between a NetScreen device and a CheckPoint 4.0 device. Otherwise, use the **subnet** switch.)

initial-contact

```
get ike initial-contact
set ike initial-contact [ all-peers | single-gateway name_str ]
unset ike initial-contact
```

initial-contact Determines how the NetScreen device performs initial contact with an IKE peer.

- Specifying **all-peers** instructs the NetScreen device to delete all SAs, then send an initial contact notification to each IKE peer.
- Specifying **single-gateway name_str** instructs the NetScreen device to delete all SAs associated with the specified IKE gateway, then send an initial contact notification.

If you specify none of the above options, the NetScreen device sends an initial contact notification to all peers during the first IKE single-user session after a system reset.

initiator-set-commit

```
get ike initiator-set-commit
set ike initiator-set-commit
unset ike initiator-set-commit
```

initiator-set-commit When the NetScreen device performs as an IKE initiator, sets the commit bit in the ISAKMP header. The party who sends the last message in the exchange does not use the new IPsec SA until it receives confirmation from the other party.

local-id

```
set ike gateway name_str { ... } local-id id_str [ ... ] { ... }
```

local-id Defines the IKE NetScreen identity of the local device. The device sends this ID to the remote gateway during IKE negotiation.

To instruct the NetScreen device to derive the IKE identity from the distinguished name in the local certificate, specify the following for **local-id** (including square brackets):

[DistinguishedName]

If there is more than one certificate on your NetScreen device, you may need to specify which certificate to use (for more information, see ["cert" on page 185](#)).

member-sa-hold-time

```
get ike member-sa-hold-time
set ike member-sa-hold-time number
unset ike member-sa-hold-time
```

member-sa-hold-time The length of time (in minutes) the device keeps an unused SA allocated for a dialup user.

nat-traversal

```
set ike gateway name_str nat-traversal udp-checksum
set ike gateway name_str nat-traversal keepalive-frequency number
unset ike gateway name_str nat-traversal [ ... ]
```

nat-traversal Enables or disables IPsec NAT Traversal, a feature that allows transmission of encrypted traffic through a NetScreen device configured for NAT. The NAT Traversal feature encapsulates ESP packets into UDP packets. This prevents the NAT device from altering ESP packet headers in transit, thus preventing authentication failure on the peer NetScreen device.

- **udp-checksum** enables the NAT-Traversal UDP checksum operation (used for UDP packet authentication).
- **keepalive-frequency** specifies the frequency (in seconds) with which the NetScreen device sends NAT-traversal keepalive messages.

Examples: The following command enables NAT traversal for a gateway named mktg:

```
set ike gateway mktg nat-traversal
```

The following command sets the Keepalive setting to 25 seconds:

```
set ike gateway mktg nat-traversal keepalive-frequency 25
```

outgoing-interface

```
set ike gateway name_str { ... } outgoing-interface interface [ ... ]
```

outgoing-interface Defines the interface through which the NetScreen device sends IKE traffic for this gateway.

Example: The following command specifies ethernet3 as the outgoing interface for an IKE gateway named Paris_Gateway at IP address 2.2.2.2. (Authentication uses a preshared key based on the word "scramble", and the Phase 1 proposals are those for the "compatible" security level for Phase 1 negotiations.)

```
set ike gateway Paris_Gateway ip 2.2.2.2 outgoing-interface ethernet3 preshare  
scramble sec-level compatible
```

p1-max-dialgrp-sessions

```
get ike p1-max-dialgrp-sessions  
set ike p1-max-dialgrp-sessions count number  
set ike p1-max-dialgrp-sessions percentage number  
unset ike p1-max-dialgrp-sessions
```

p1-max-dialgrp-sessions Displays the allowed concurrent Phase 1 negotiations for dialup groups.

p1-proposal

```
get ike p1-proposal name_str
set ike p1-proposal name_str [ ... ] { ... }
unset ike p1-proposal name_str
```

p1-proposal

Names the IKE Phase 1 proposal, which contains parameters for creating and exchanging session keys and establishing Phase 1 security associations.

- **dsa-sig** | **rsa-sig** | **preshare** Specifies the method to authenticate the source of IKE messages. **preshare** refers to a preshared key, which is a key for encryption and decryption that both participants have before beginning tunnel negotiations. **rsa-sig** and **dsa-sig** refer to two kinds of digital signatures, which are certificates that confirm the identity of the certificate holder. (The default method is **preshare**.)
- **group1** | **group2** | **group5** Identifies the Diffie-Hellman group, a technique that allows two parties to negotiate encryption keys over an insecure medium; such as, the Internet. Group2 is the default group.
- **esp** Specifies Encapsulating Security Payload protocol, which provides encryption and authentication.
- **des** | **3des** | **aes128** | **aes192** | **aes256** Specifies the encryption algorithm.
- **md5** | **sha-1** Specifies the authentication (hashing) algorithm used in ESP protocol. The default algorithm is SHA-1, the stronger of the two algorithms.
- The following parameters define the elapsed time between each attempt to renegotiate a Phase 1 security association. The minimum allowable lifetime is 180 seconds. The default lifetime is 28800 seconds.
 - **days** *number*
 - **hours** *number*
 - **minutes** *number*
 - **seconds** *number*

Example: The following command defines a Phase 1 proposal named sf1.

- Preshared key and a group 1 Diffie-Hellman exchange
- Encapsulating Security Payload (ESP) protocol using the 3DES and MD5 algorithms
- Lifetime of 3 minutes

```
set ike p1-proposal sf1 preshare group1 esp 3des md5 minutes 3
```

p1-sec-level

```
get ike p1-sec-level
```

p1-sec-level Displays the predefined IKE Phase 1 proposals in descending order of security level.

p2-sec-level

```
get ike p2-sec-level
```

p2-sec-level Displays the predefined IKE Phase 2 proposals in descending order of security level.

p2-proposal

```
get ike p2-proposal name_str
set ike p2-proposal name_str [ ... ] { ... }
set ike p2-proposal name_str
```

p2-proposal Names the IKE Phase 2 proposal. This proposal defines parameters for creating and exchanging a session key to establish a security association (SA).

- **group1 | group2 | group5 | no-pfs** Defines how the NetScreen device generates the encryption key. Perfect Forward Secrecy (PFS) is a method for generating each new encryption key independently from the previous key. Selecting **no-pfs** turns this feature off, so IKE generates the Phase 2 key from the key generated in the Phase 1 exchange. If you specify one of the Diffie-Hellman groups, IKE automatically uses PFS when generating the encryption key. The default is Group 2.

- **ah | esp** In a Phase 2 proposal, identifies the IPSec protocol.
 - **esp** [**des** | **3des** | **aes128** | **aes192** | **aes256**] Specifies Encapsulating Security Payload (ESP) protocol, which provides both encryption and authentication. Specifies the encryption algorithm used in ESP protocol. (The default protocol is **des**.)
 - **ah** Specifies Authentication Header (AH) protocol, which provides authentication only.
- **md5 | null | sha-1** Specifies the authentication (hashing) algorithm used in ESP or AH protocol. The default algorithm is MD5 for non-FIPS mode, and SHA is the default for FIPS mode. The **null** switch specifies no authentication.
- The following parameters define the elapsed time between each attempt to renegotiate a security association. The minimum allowable lifetime is 180 seconds. The default lifetime is 28800 seconds.
 - **days** *number*
 - **hours** *number*
 - **minutes** *number*
 - **seconds** *number*
- **kbytes** *number* Indicates the maximum allowable data flow in kilobytes before NetScreen renegotiates another security association. The default value is **0** (infinity).

Example: The following command specifies Phase 2 proposal g2-esp-3des-null.

- Group 2 Diffie-Hellman exchange
- ESP using 3DES without authentication
- Lifetime of 15 minutes

```
set ike p2-proposal g2-esp-3des-null group2 esp 3des null minutes 15
```

policy-checking

```
get ike policy-checking
set ike policy-checking
unset ike policy-checking
```

policy-checking Checks to see if the policies of the two peers match before establishing a connection. Use policy checking when configuration on the peer gateways support multiple tunnels. Otherwise, the IKE session fails. You can disable policy checking when only one policy is configured between two peers.

preshare

```
set ike p1-proposal name_str preshare [ ... ]
```

preshare Directs the device to use preshared key authentication for IKE Phase 1 negotiation. In this mode, both peer devices use a shared password to generate a encryption and decryption key.

```
set ike gateway name_str { ... } [ ... ] preshare key_str
```

preshare Specifies the Preshared key (*key_str*) used in the Phase 1 proposal. (If you use an RSA- or DSA-signature in the Phase 1 proposal, do not use this option).

Example: For an example of this option, see ["Setting Up a Policy-Based VPN Tunnel" on page 202](#).

preshare-gen

exec ike preshare-gen *name_str usr_str*

preshare-gen

Generates an individual preshared key for a remote dialup user associated with a Group IKE ID user. The NetScreen device generates each preshared key from a seed value (specified in the command **set ike gateway**). After the device generates the preshared key, you can use it to set up a configuration for the remote user. (Remove any spaces.)

- *name_str* is the IKE gateway name. To create such a gateway, use the **set ike gateway** *name_str* command.
- *usr_str* is the full IKE ID of an individual user, which belongs to a Group IKE ID user. To create such a user, use the **set user** *name_str ike-id* command. The Group IKE ID user must be associated with a dialup user group to support a group of users.

Example: The following commands create a single group IKE ID user and assign the user to a dialup user group. Then they create VPNs and policies that allow dialup users with matching partial IKE ID values to establish secure communication through the NetScreen device.

- The name of the group IKE ID user is User1, with partial IKE identity of acme.com.
- The number of dialup users that can share this user's IKE identity is 10.
- The dialup user group is Office_1.
- The seed value for creating the preshared key is jk930k.
- The Phase 1 IKE gateway defined for the server side is Corp_GW.
- The Phase 2 VPN defined for the server side is Corp_VPN.
- The Phase 1 IKE gateway defined for the client side is Office_GW.
- The Phase 2 VPN defined for the client side is Office_VPN.
- The individual user's full IKE identity is chris@acme.com.
- The trusted server that dialup users access from the outside is a Web server with IP address 1.1.110.200.

```
set user User1 ike-id u-fqdn acme.com share-limit 10
set user-group Office_1 user User1
set ike gateway Corp_GW dialup Office_1 aggressive seed-preshare jk930k
    proposal pre-g2-3des-md5
set vpn Corp_VPN gateway Corp_GW tunnel proposal g2-esp-3des-md5
set address trust http_server 1.1.110.200 255.255.255.255
set policy incoming "dial-up vpn" http_server any tunnel vpn Corp_VPN
```

To generate the preshared key for [chris@acme.com](#):

```
exec ike preshare-gen Corp_GW chris@acme.com
```

Note: For this example, assume that this command generates `c5d7f7c1806567bc57d3d30d7bf9b93baa2adcc6`.

On the client side:

```
set ike gateway Office_GW address 10.1.10.10 aggressive local-id chris@acme.com
    preshare c5d7f7c1806567bc57d3d30d7bf9b93baa2adcc6 proposal pre-g2-3des-md5
set vpn Office_VPN gateway Office_GW tunnel proposal g2-esp-3des-md5
set address untrust http_server 1.1.110.200
set policy outgoing "inside any" http_server any tunnel vpn Office_VPN
```

proposal

```
set ike gateway name_str { ... } [ ... ] proposal name_str1
    [ name_str2 ] [ name_str3 ] [ name_str4 ]
```

proposal Specifies the name (*name_str*) of a proposal. You can specify up to four Phase 1 proposals.

Example: For an example of this option, see ["Setting Up a Policy-Based VPN Tunnel"](#) on page 202.

respond-bad-spi

```
get ike respond-bad-spi
set ike respond-bad-spi [ number ]
unset ike respond-bad-spi
```

respond-bad-spi Responds to packets with bad security parameter index (SPI) values. The specified *number* value is the number of times to respond to bad SPIs per gateway.

responder-set-commit

```
get ike responder-set-commit
set ike responder-set-commit
unset ike responder-set-commit
```

responder-set-commit Directs the NetScreen device to set the commit bit in the ISAKMP header when the device acts as an IKE responder. The peer that sends the last message in the exchange does not use the new IPsec SA until it receives information from the other peer.

sec-level

```
set ike gateway name_str { ... } [ ... ] sec-level { ... }
```

sec-level Specifies which pre-defined security proposal to use for IKE. The **basic** proposal provides basic-level security settings. The **compatible** proposal provides the most widely-used settings. The **standard** proposal provides settings recommended by Juniper Networks.

Example: The following command specifies the pre-defined security proposal **compatible**:

```
set vpn Corp_VPN gateway Corp_GW sec-level compatible
```

seed-preshare

```
set ike gateway name_str { ... } [ ... ] seed-preshare key_str
```

seed-preshare Specifies a seed value (*key_str*) for a user group with Preshared Key configurations. Such a configuration performs IKE authentication for multiple dialup users, each with an individual preshared key, without having a separate configuration for each user. Instead, use the seed to generate the preshared key with the **exec ike preshare-gen** command.

Example: The following commands configure IKE authentication for multiple dialup users in a user group:

- Interface ethernet1 bound to the Trust zone and interface ethernet3 bound to the Untrust zone
- Dialup user named User2, placed in a user group named office_2
- Gateway configuration for office_2, with a preshared key seed value of jk930k
- Security policy for all dialup users with the partial IKE identity specified for User2

```
set interface ethernet1 zone trust
set interface ethernet1 ip 10.1.1.1/24
set interface ethernet3 zone untrust
set interface ethernet3 ip 1.1.1.1/24
set address trust web1 10.1.1.5/32
set user User2 ike-id u-fqdn juniper.net share-limit 10
set user-group office_2 user User2
set ike gateway Corp_GW dialup office_2 aggressive seed-preshare jk930k
    sec-level compatible
set vpn Corp_VPN gateway Corp_GW sec-level compatible
set policy top from untrust to trust "Dial-Up VPN" web1 http tunnel vpn Corp_VPN
save
```

single-ike-tunnel

```
set ike single-ike-tunnel name_str  
unset ike single-ike-tunnel name_str
```

single-ike-tunnel Specifies a single Phase 2 SA for all policies to a particular remote peer gateway.

Example: The following command specifies a Phase 2 SA for all policies to the peer gateway gw1:

```
set ike single-ike-tunnel gw1
```

soft-lifetime-buffer

```
get ike soft-lifetime-buffer  
set ike soft-lifetime-buffer number
```

soft-lifetime-buffer Sets a time interval (in seconds) before the current IPSec SA key lifetime expires. When this interval is reached, the device initiates the rekeying operation.

xauth

```
set ike gateway name_str xauth  
unset ike gateway xauth
```

xauth Enables XAuth authentication for the specified IKE gateway configuration.

xauth bypass-auth

```
set ike gateway name_str xauth bypass-auth
```

bypass-auth Instructs the NetScreen device, acting as an XAuth server, to perform only XAuth mode-config, which assigns the XAuth client with an IP address, and DNS and WINS server settings. The XAuth client is not required to authenticate him or herself.

xauth client

```
set ike gateway name_str xauth client { any | chap | securid } username name_str
password string
```

client	Specifies that the NetScreen device is an XAuth client. You can specify the following authentication types: <ul style="list-style-type: none">• any Instructs the device to allow any authentication type.• chap Instructs the device to allow Challenge Handshake Authentication Protocol (CHAP) only.• securid Instructs the device to allow authentication via SecurID only.
username	Specifies the username for the XAuth client to use on the XAuth server.
password	Specifies the password for the XAuth client to use on the XAuth server.

Example: The following example configures an XAuth client.

- Gateway kg1
- Any authentication type allowed
- Username kgreen and password pubs123

```
set ike gateway kg1 xauth client any username kgreen password pubs123
```

xauth server

```
set ike gateway name_str xauth server name_str
set ike gateway name_str xauth server name_str [ chap ] [ query-config ] [ user
name_str | user-group name_str ]
unset ike gateway xauth
```

server	Specifies the object name of the external server that performs the XAuth authentication. <ul style="list-style-type: none">• chap Instructs the device to use Challenge Handshake Authentication Protocol (CHAP).• query-config Instructs the device to query the client configuration from the server.
---------------	--

- **user** *name_str* Enables XAuth authentication for an individual user.
- **user-group** *name_str* Enables XAuth authentication for the users in a XAuth user group.

Defaults

Main mode is the default method for Phase 1 negotiations.

The default time intervals before the NetScreen mechanism renegotiates another security association are 28,800 seconds in a Phase 1 proposal, and 3600 seconds in a Phase 2 proposal.

The default ID mode is subnet. (Changing the ID mode to IP is only necessary if the data traffic is between two security gateways, one of which is a CheckPoint 4.0 device.)

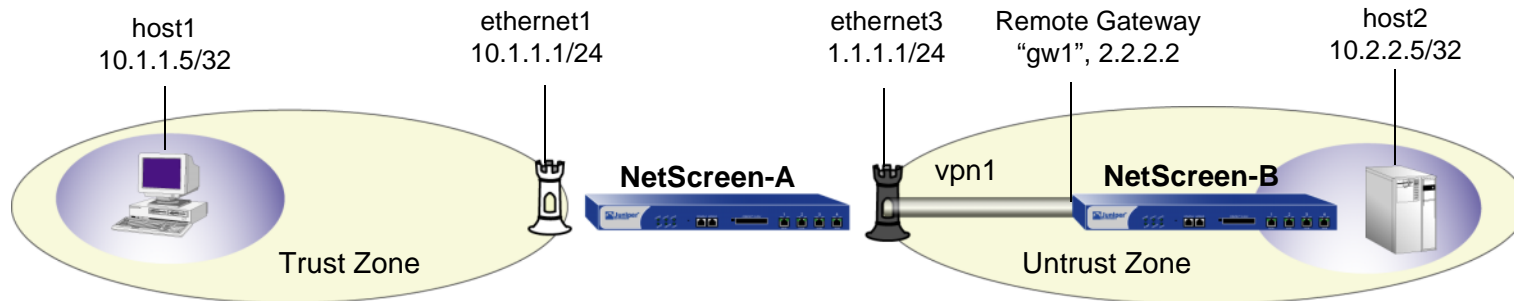
The default soft-lifetime-buffer size is 10 seconds.

By default, the single-ike-tunnel flag is not set.

By default, the commit bit is not set when initiating or responding to a Phase 2 proposal.

Setting Up a Policy-Based VPN Tunnel

Creating a policy-based VPN tunnel for a remote gateway with a static IP address requires up to seven steps. To set up NetScreen-A at one end of a VPN tunnel for bidirectional traffic, follow the steps below.



1. Bind interfaces to zones and assign them IP addresses:

```
set interface ethernet1 zone trust  
set interface ethernet1 ip 10.1.1.1/24  
set interface ethernet3 zone untrust  
set interface ethernet3 ip 1.1.1.1/24
```
2. Set the addresses for the end entities beyond the two ends of the VPN tunnel:

```
set address trust host1 10.1.1.5/32  
set address untrust host2 10.2.2.5/32
```
3. Define the IKE Phase 1 proposal and Phase 2 proposal. If you use the default proposals, you do not need to define Phase 1 and Phase 2 proposals.
4. Define the remote gateway:

```
set ike gateway gw1 address 2.2.2.2 main outgoing-interface ethernet3 preshare  
netscreen proposal pre-g2-3des-sha
```
5. Define the VPN tunnel as AutoKey IKE:

```
set vpn vpn1 gateway gw1 proposal g2-esp-des-md5
```


6. Set a default route (both the Trust and Untrust zones are in the trust-vr routing domain):
`set vrouter trust-vr route 0.0.0.0/0 interface ethernet3 gateway 1.1.1.250`
7. Set outbound and inbound policies:
`set policy from trust to untrust host1 host2 any tunnel vpn vpn1`
`set policy from untrust to trust host2 host1 any tunnel vpn vpn1`

The procedure for setting up a VPN tunnel for a dialup user with IKE also constitutes up to seven steps.

1. Bind interfaces to zones and assign them IP addresses.
2. Define the protected address that you want the dialup user to be able to access through the tunnel. (See the **set address** command.)
3. Define the user as an IKE user. (See the **set user** command.)
4. Define the IKE Phase 1 proposal, Phase 2 proposal, and remote gateway. (Note: If you use the default proposals, you do not need to define a Phase 1 or Phase 2 proposal.)
5. Define the VPN tunnel as AutoKey IKE. (See the **set vpn** command.)
6. Set a default route (both the Trust and Untrust zones are in the trust-vr routing domain).
7. Define an incoming policy, with **Dial-Up VPN** as the source address and the VPN tunnel you configured in step 5.

ike-cookie

Description: Use the **ike-cookie** command to remove IKE-related cookies from the NetScreen device.

Syntax

clear

```
clear [ cluster ] ike-cookie { all | ip_addr }
```

Keywords and Variables

Variable Parameter

```
clear cluster ike-cookie ip_addr  
clear ike-cookie ip_addr
```

ip_addr Directs the NetScreen device to remove cookies based on a IP address (*ip_addr*).

Example: The following command removes all cookies based on the IP address 10.1.10.10:

```
clear ike-cookie 10.1.10.10
```

all

```
clear cluster ike-cookie all  
clear ike-cookie all
```

all Directs the NetScreen device to remove all cookies.

cluster

```
clear cluster ike-cookie all  
clear cluster ike-cookie ip_addr
```

cluster Propagates the **clear** operation to all other devices in a NSRP cluster.

interface

Description: Use the **interface** commands to define or display interface settings for a NetScreen device.

Interfaces are physical or logical connections that handle network, virtual private network (VPN), High Availability (HA), and administrative traffic. For a description of the interfaces you can configure on a NetScreen device, see ["Interface Names" on page A-1](#).

Syntax

get

```
get interface interface
[
  dhcp
  {
    client |
    relay |
    server { ip { allocate | idle } | option }
  } |
  dip |
  mip |
  protocol { ospf | rip } |
  screen |
  secondary [ ip_addr ] |
  track-ip [ ip ]
]
```

set (Layer-3 Interfaces)

```
set interface interface
{
  bandwidth number |
  dip id_num
  {
    ip_addr1 [ ip_addr2 ] |
    shift-from ip_addr3 [ to ip_addr4 [ ip_addr5 ] ]
  }
  [ fix-port ] |
  [ ext ip ip_addr/mask ] dip id_num
  {
    ip_addr1 |
    shift-from ip_addr2 to ip_addr3
  } [ ip_addr4 ] [ fix-port ] |
  gateway ip_addr [ no-default-interface ] |
  group |
  ip ip_addr/mask { tag id_num } |
  manage { ident-reset | nsmgmt | ping | snmp | ssh | ssl | telnet | web } |
  manage-ip ip_addr |
  mip ip_addr host ip_addr [ netmask mask ] [ vrouter name_str ] |
  mtu number |
  nat |
  phy
  {
    auto |
    full { 10mb | 100mb } |
    half { 10mb | 100mb } |
    holddown number |
    link-down
  } |
  route |
  route-deny |
  tag id_num zone zone /
```

```
vip ip_addr [ + ] port_num [ name_str ip_addr [ manual ] ] |
webauth |
webauth-ip ip_addr |
zone zone
}
```

set (Layer-2 Interfaces)

```
set interface interface
{
  manage { ident-reset | nsmgmt ping | nmp | ssh | ssl | telnet | web } |
  phy
  {
    auto |
    full { 10mb | 100mb } |
    half { 10mb | 100mb } |
    holddown number |
    link-down
  } |
  webauth
}
```

set (DHCP Relay/Server)

```
set interface interface dhcp
{
  relay { server-name { name_str | ip_addr } | service | vpn } |
  server
  {
    enable | auto | disable
    ip ip_addr { mac mac_addr | to ip_addr } |
    option
    {
      dns1 | dns2 | dns3 | gateway | news | nis1 | nis2 | pop3 | smtp
      { ip_addr } |
    }
  }
}
```

```
        domainname name_str |
        lease number |
        netmask mask |
        nistag name_str |
        wins1 ip_addr |
        wins2 ip_addr
    } |
    service
}
```

set (DHCP Client)

```
set interface interface dhcp client
{
    enable |
    settings
    {
        autoconfig |
        lease number |
        server ip_addr |
        update-dhcpserver |
        vendor id_str
    }
}
```

set (High Availability)

```
set interface { ha | ha1 | ha2 }
{
    bandwidth number |
    phy
    {
        auto |
        full { 10mb | 100mb } |
    }
}
```

```
    half { 10mb | 100mb } |  
    holddown number |  
    link-down  
  } |  
}
```

set (IP Tracking)

```
set interface interface track-ip  
[  
  dynamic |  
  ip ip_addr  
  [  
    interval number |  
    threshold number |  
    weight number  
  ] |  
  threshold number  
]
```

set (Loopback Interface)

```
set interface interface loopback-group interface
```

set (BGP)

```
set interface interface protocol bgp
```


set (OSPF)

```
set interface interface protocol ospf
{
  area { ip_addr | number } |
  authentication
  {
    active-md5-key-id id_num |
    md5 key_str [ key-id id_num ] |
    password pswd_str
  } |
  cost number |
  dead-interval number |
  enable |
  hello-interval number |
  link-type point-to-point |
  neighbor-list number1 [ number2 [ number3 [ number4 ] ] ] |
  passive |
  priority number |
  retransmit-interval number |
  transit-delay number
}
```

set (RIP)

```
set interface interface protocol rip
[
  authentication
  {
    active-md5-key-id id_num |
    md5 key_str [ key-id id_num ] |
    password pswd_str
  } |
  enable |
  metric number |
]
```

```
passive-mode |  
route-map name_str |  
split-horizon [ poison-reverse ]  
]
```

set (Tunnel)

```
set interface tunnel.number  
{  
  dip id_num  
  {  
    ip_addr1  
      [ ip_addr2 ] [ fix-port ] |  
    shift-from ip_addr3  
  } |  
  [ ext ip ip_addr/mask ] dip id_num  
  {  
    ip_addr1  
      [ ip_addr2 ] [ fix-port ] |  
    shift-from ip_addr3  
  } |  
  ip ip_addr/mask |2  
  loopback-group |  
  manage-ip ip_addr |  
  mip ip_addr host ip_addr  
    [ netmask mask [ vrouter name_str ] ] |  
  nhtb ip_addr vpn tunn_str |  
  protocol { bgp | ospf } |  
  route-deny |  
  zone name_str  
}
```

-
2. Use the **IP** option only after adding the tunnel to a specific zone.

unset (vlan1 interface)

```
unset interface interface
{
  broadcast [ arp [ trace-route ] ] |
  bypass-non-ip | | bypass-non-ip-all |
  bypass-others-ipsec
}
```

unset (DHCP Relay/Server)

```
unset interface interface dhcp
{
  relay { server-name { name_str | ip_addr } |
  server
  {
    ip { ip_addr / all } |
    option
    {
      dns1 | dns2 | dns3 | gateway | news | nis1 | nis2 | pop3 | smtp |
      domainname |
      lease |
      netmask |
      nistag |
      wins1 |
      wins2
    }
    service
  }
}
```

unset (DHCP Client)

```
unset interface interface dhcp client
{
  enable |
  settings
  {
    autoconfig |
    lease |
    server |
    update-dhcpserver |
    vendor
  }
}
```

unset

```
unset interface interface
{
  bandwidth |
  [ ext ip ip_addr mask ] dip number |
  group |
  ip [ ip_addr ] |
  loopback-group interface |
  manage { ident-reset | nsmgmt | ping | snmp | ssh | ssl | telnet | web } |
  manage-ip |
  mip ip_addr1 host ip_addr2 [ netmask mask ] |
  mtu |
  nhtb ip_addr |
  phy
  {
    auto |
    full { 10mb | 100mb } |
    half { 10mb | 100mb } |
    holddown |
  }
}
```

```
    link-down
  }
protocol ospf
{
  area |
  authentication { active-md5-key-id | md5 [ key-id id_num ] } |
  cost |
  dead-interval |
  enable |
  hello-interval |
  link-type point-to-point |
  neighbor-list |
  passive |
  priority |
  retransmit-interval |
  transit-delay
} |
protocol rip
{
  authentication { active-md5-key-id | md5 [ key-id id_num ] } |
  enable |
  metric |
  passive-mode |
  route-map |
  split-horizon [ poison-reverse ]
} |
route-deny |
track-ip
[
  dynamic |
  ip ip_addr [ interval | threshold | weight ] |
  threshold
]
vlan trunk |
```

```
webauth |  
webauth-ip |  
zone  
}
```

unset (DHCP)

```
unset interface interface dhcp  
{  
  relay { server-name { name_str | ip_addr } | service | vpn }  
  server  
  {  
    ip ip_addr  
    option  
    {  
      dns1 | dns2 | dns3 |  
      domainname |  
      gateway }  
      lease |  
      news |  
      netmask |  
      nis1 | nis2  
      nistag |  
      pop3 |  
      smtp |  
      wins1 |  
      wins2  
    }  
    service  
  }  
}
```

Keywords and Variables

Variable Parameter

```
get interface interface [ ... ]  
set interface interface { ... } [ ... ]
```

interface The name of the interface. For more information on interface names, see ["Interface Names" on page A-I](#).

Example: The following command specifies the IP address of a remote gateway peer (1.1.1.25) for the ethernet4 interface:

```
set interface ethernet4 gateway 1.1.1.25
```

bandwidth

```
set interface interface bandwidth number  
unset interface interface bandwidth
```

bandwidth The guaranteed maximum bandwidth in kilobits per second for all traffic traversing the specified interface.

Example: The following command specifies bandwidth of 10,000 kilobits per second for interface **ethernet4**:

```
set interface ethernet4 bandwidth 10000
```

broadcast

```
set interface interface broadcast { flood | arp [ trace-route ] }  
unset interface interface broadcast [ arp [ trace-route ] ]
```

- broadcast** (vlan1 interface only.) Controls how the NetScreen device determines reachability of other devices while the device is in transparent (L2) mode.
- **flood** Instructs the NetScreen device to flood frames received from an unknown host out to all interfaces that are in transparent mode. In the process, the device might attempt to copy frames out of ports that cannot access the destination address, thus consuming network bandwidth.
 - **arp [trace-route]** Instructs the NetScreen device to generate an Address Resolution Protocol (ARP) broadcast. If the broadcast finds the unknown destination IP address, the device loads its ARP table with the appropriate MAC address and interface. The device uses this entry to reach the destination device directly, and only sends frames through the correct port, thus saving bandwidth. Generating the initial ARP can cause delay, but only for the first frame.

Example: The following command instructs the NetScreen device to generate an Address Resolution Protocol (ARP) broadcast:

```
set interface vlan1 broadcast arp
```

bypass-non-ip

```
set interface interface bypass-non-ip  
unset interface interface bypass-non-ip
```

- bypass-non-ip** (vlan1 interface only.) Allows non-IP traffic (such as IPX) to pass through a NetScreen device running in Transparent mode. (ARP is a special case for non-IP traffic. It is always passed, even if when this feature is disabled.) (The bypass-non-ip-all does not allow non-broadcast, non-multicast traffic.)

bypass-non-ip-all

```
set interface interface bypass-non-ip-all
unset interface interface bypass-non-ip-all
```

bypass-non-ip-all (**vlan1** interface only.) Allows non-broadcast, non-multicast, and non-IP traffic to pass through a NetScreen device running in Transparent mode. (ARP is a special case for non-IP traffic. It is always passed, even if when this feature is disabled.)

bypass-others-ipsec

```
set interface interface bypass-others-ipsec
unset interface interface bypass-others-ipsec
```

bypass-others-ipsec (**vlan1** interface only.) Openly passes all IPSec traffic through a NetScreen device in Transparent mode. The NetScreen device does not act as a VPN tunnel gateway but passes the IPSec packets onward to other gateways.

dhcp client

```
set interface interface dhcp client
{
  enable |
  settings
  {
    autoconfig |
    lease number |
    server ip_addr |
    update-dhcpserver | vendor id_str } }
```

dhcp client

Configures an interface for DHCP client services.

- **enable** Enables DHCP client services for the interface.
- **settings** Configures DHCP parameters for the interface.
 - **autoconfig** Enables automatic configuration after device power-up.
 - **lease *number*** Sets the default lease time (in minutes).
 - **server *ip_addr*** Specifies the IP address of the DHCP server.
 - **update-dhcpserver** Forwards TCP/IP settings from the DHCP client module on the specified interface to the DHCP server module on the default interface in the Trust zone. Note: On devices that can have multiple interfaces bound to the Trust zone, the default interface is the first interface bound to that zone and assigned an IP address.
 - **vendor *id_str*** Specifies the DHCP vendor by ID.

Examples: The following command configures interface **ethernet3** to perform automatic DHCP configuration after device power-up:

```
set interface ethernet3 dhcp client settings autoconfig
```

The following command enables (the forwarding of TCP/IP settings from the DHCP client module on the Untrust interface to the DHCP server module on the Trust zone interface):

```
set interface untrust dhcp client settings update-dhcpserver
```

dhcp relay

```
get interface interface dhcp relay
set interface interface dhcp relay
    { server-name name_str | service | vpn }
unset interface interface dhcp relay { server-name { name_str | ip_addr } | service
    | vpn }
```

- dhcp relay** Configures the NetScreen interface such that the NetScreen device can serve as a DHCP relay agent.
- **server-name** *name_str* Defines the domain name of the external DHCP server from which the NetScreen device receives the IP addresses and TCP/IP settings that it relays to hosts on the LAN.
 - **service** Enables the NetScreen device to act as a DHCP server agent through the interface.
 - **vpn** Allows the DHCP communications to pass through a VPN tunnel. You must first set up a VPN tunnel between the NetScreen device and the external DHCP server.

The relay does not coexist with the DHCP server (OK with the client).

Example: The following configures interface **ethernet4** to use an external DHCP server at IP address 1.1.1.10:

```
set interface ethernet4 dhcp relay server-name 1.1.1.10
```

dhcp server

```
set interface interface dhcp server { ... }
unset interface interface dhcp server { ... }
```

- dhcp server** Makes the NetScreen interface work as a DHCP server.
- **auto** Instructs the NetScreen device to check to see if there is a DHCP server already running on the network. If there is such a server, the DHCP server on the NetScreen device is disabled. If there is no DHCP server running on the network, the DHCP server on the NetScreen device is enabled. This is the default mode.
 - **disable** Causes the DHCP server to always be off.

- **enable** Causes the DHCP server to always be on. The DHCP server on the NetScreen device always starts when the device is powered on.
- **ip** *ip_addr* { **mac** *mac_addr* | **to** *ip_addr* } Specifies either a specific IP address that is assigned to a host or the lower end of a range of IP addresses to use when the DHCP server is filling client requests.
 - **mac** This option allows you to statically assign an IP address to the host that is identified by the specified MAC address. The host is always assigned the specified IP address.
 - **to** Defines the upper end of a range of IP addresses to use when the DHCP server is filling client requests. The IP pool can support up to 255 IP addresses. The IP address must be in the same subnet as the interface IP or the DHCP gateway.
- **option** Specifies the DHCP server options for which you can define settings.
 - **dns1** *ip_addr* | **dns2** *ip_addr* | **dns3** *ip_addr* Defines the IP addresses of the primary, secondary, and tertiary Domain Name Service (DNS) servers.
 - **gateway** *ip_addr* Defines the IP address of the gateway to be used by the clients. The IP address must be in the same subnet as the interface IP or the DHCP gateway.
 - **news** *ip_addr* Specifies the IP address of a news server to be used for receiving and storing postings for news groups.
 - **nis1** *ip_addr* | **nis2** *ip_addr* Defines the IP addresses of the primary and secondary NetInfo[®] servers, which provide the distribution of administrative data within a LAN.
 - **pop3** *ip_addr* Specifies the IP address of a Post Office Protocol version 3 (POP3) mail server.
 - **smtp** *ip_addr* Defines the IP address of a Simple Mail Transfer Protocol (SMTP) mail server.
 - **domainname** *name_str* Defines the registered domain name of the network.
 - **lease** *number* Defines the length of time, in minutes, for which an IP address supplied by the DHCP server is leased. For an unlimited lease, enter **0**.
 - **netmask** *ip_addr* Defines the netmask of the gateway. The IP address must be in the same subnet as the interface IP or the DHCP gateway.
 - **nistag** *string* Defines the identifying tag used by the Apple[®] NetInfo database.

- **wins1** *ip_addr* | **wins2** *ip_addr* Specifies the IP address of the primary and secondary Windows Internet Naming Service (WINS) servers.
- **service** Enables the NetScreen device to act as a DHCP server agent through the interface.

The server does not coexist with the DHCP relay (OK with the client).

Example: The following command configures the NetScreen device to act as a DHCP server agent through the interface **ethernet4**:

```
set interface ethernet4 dhcp server service
```

dip

```
set interface interface dip id_num ip_addr1 [ ip_addr2 ] [ fix-port ]  
set interface interface dip id_num shift-from ip_addr3  
unset interface interface dip id_num
```

dip

Sets a Dynamic IP (DIP) pool. Each DIP pool consists of a range of addresses. The NetScreen device can use the pool to dynamically or deterministically allocate source addresses when the device applies source address translation (NAT-src) to packets traversing the specified interface. This is useful when you need to translate non-routable local IP source addresses into routable addresses for outgoing packet. The keywords and variables for the **dip** option are as follows:

- *id_num* Identifies the DIP pool.
- The first IP address *ip_addr1* represents the start of the IP address range. (A DIP pool can consist of a single IP address, or a range of addresses.) The second IP address *ip_addr2* represents the end of the IP address range.
- **shift-from** *ip_addr3* Defines a one-to-one mapping from an original source IP address to a translated source IP address for a range of IP addresses starting from *ip_addr3*. Such a mapping ensures that the NetScreen device always translates a particular source IP address from within that range to the same translated address within a DIP pool.

Be sure to exclude the following IP addresses from a DIP pool:

- The WebUI management IP address
- The interface and gateway IP addresses
- Any Virtual IP (VIP) and Mapped IP (MIP) addresses

Example1: The following commands allow local hosts in a non-routable subnet to communicate over a public WAN infrastructure. The NetScreen device uses a DIP pool to dynamically allocate routable source addresses to packets sent from the local hosts to remote hosts.

- Local unroutable subnet 10.1.23.1/24
- Remote unroutable subnet 10.100.2.75/24
- DIP ID number 10, with address range from 2.1.10.2 through 2.1.10.36

```
unset interface ethernet2 ip
unset interface ethernet2 zone
unset interface ethernet3 ip
unset interface ethernet3 zone
```

```
set interface ethernet2 zone trust
set interface ethernet2 ip 10.1.23.1/24
```

```
set interface ethernet3 zone untrust
set interface ethernet3 ip 2.1.10.1/24
set interface ethernet3 dip 10 2.1.10.2 2.1.10.36
```

```
set address trust Local_Hosts 10.1.23.1/24
set address untrust Remote_Hosts 10.100.2.75/24
```

```
set policy from trust to untrust Local_Hosts Remote_Hosts http nat dip 10 permit
```

ext ip

```
set interface interface ext ip ip_addr/mask dip number { ... }  
unset interface interface ext ip ip_addr/mask dip number
```

- ext ip** The **ext ip** *ip_addr* option configures a DIP in a different subnet from the interface's subnet. For example, an interface could have IP address 1.2.10.1/24, and the extended DIP could be 2.2.3.1/24.
- **dip** *id_num* Sets a Dynamic IP (DIP) pool. See ["dip" on page 223](#).
 - **fix-port** Keeps the original source port number in the packet header. Does not apply the Port Address Translation (PAT).

Example: The following command creates an address (1.1.100.110) in a DIP (ID 10) for interface **ethernet3** (IP address 10.1.10.10):

```
set interface ethernet3 ext ip 10.1.10.10/24 dip 10 1.1.100.110
```

gateway

```
set interface interface gateway ip_addr [ no-default-route ]  
unset interface interface gateway
```

- gateway** The IP address for the default gateway to which the NetScreen device forwards packets that are destined for networks beyond the immediate subnet of the specified interface. The **no-default-route** switch specifies that there is no default route for this gateway.

Example: The following command specifies the IP address of a remote gateway peer (1.1.10.10) for the **ethernet4** interface:

```
set interface ethernet4 gateway 1.1.10.10
```

ip

```
set interface interface ip ip_addr/mask [ secondary ]  
unset interface interface ip ip_addr
```

ip The IP address *ip_addr* and netmask *mask* for the specified interface or subinterface. The **secondary** switch specifies that the IP address is a secondary address.

Example: The following commands create logical interface ethernet3/1.2, bind it to the Trust zone, and assign it IP address 10.1.40.3/24:

```
set interface ethernet3/1.2 zone trust  
set interface ethernet3/1.2 ip 10.1.40.3/24
```

loopback-group

```
set interface interface1 loopback-group loopback.n  
unset interface interface1 loopback-group loopback.n
```

loopback-group Adds a specified interface (*interface1*) to the loopback group for a designated loopback interface (loopback.*n*). All members in the loopback group can share the MIP (Mapped IP) and DIP (Dynamic IP) definitions assigned to the loopback interface itself.

Example: The following commands add interfaces ethernet1 and ethernet2 to the loopback group for loopback.1, and then assign a MIP to loopback.1. This allows both ethernet1 and ethernet2 to use the assigned MIP.

```
set interface ethernet1 loopback-group loopback.1  
set interface ethernet2 loopback-group loopback.1  
set int loopback.1 mip 1.1.1.1 host 10.1.1.8 netmask 255.255.255.0
```


manage

```
set interface interface manage
  { ident-reset | nsmgmt | ping | snmp | ssh | ssl | telnet | web }
unset interface interface manage
  { ident-reset | nsmgmt | ping | snmp | ssh | ssl | telnet | web }
```

manage

Enables or disables monitoring and management capability through the interface.

- **ident-reset** Directs the NetScreen device to send a TCP Reset announcement, in response to an IDENT request, to port 113.
- **nsmgmt** Enables or disables NSM (Juniper Networks NetScreen-Security Manager) on the interface. NSM is an enterprise-level management application that configures NetScreen devices from remote hosts. For more information, see ["nsmgmt" on page 286](#).
- **ping** Enables (or disables) pinging through the interface.
- **snmp** Enables (or disables) SNMP management through the interface.
- **ssh** Enables (or disables) SSH management through the interface.
- **ssl** Enables (or disables) SSL management through the interface.
- **telnet** Enables (or disables) telnet management through the interface.
- **web** Enables (or disables) web management through the interface.

Example: The following command enables management of SSH through interface **ethernet3**:

```
set interface ethernet3 manage ssh
```

manage-ip

```
set interface interface manage-ip ip_addr  
unset interface interface manage-ip
```

manage-ip Defines the Manage IP address for the specified physical interface. External applications such as Telnet or WebUI can use this address to configure and monitor the NetScreen device. (This address must be in the same subnet as the interface IP address.)

Example: The following commands bind interface **ethernet4/1** to the Trust zone, then set the Manage IP address to 10.1.10.10:

```
set interface ethernet4/1 zone trust  
set interface ethernet4/1 manage-ip 10.1.10.10
```

mip

```
set interface interface mip ip_addr1 host ip_addr2 [ vrouter vrouter ] [ netmask  
    mask ]  
unset interface interface mip ip_addr1 [ netmask mask ]
```

mip Defines a Mapped IP (MIP) address for the NetScreen interface. The device directs traffic sent to the MIP (*ip_addr1*) to the host with the IP address *ip_addr2*. Setting a MIP for an interface in any zone generates a book entry for the MIP in the Global zone address book. The Global zone address book keeps all the MIPs of all interfaces, regardless of the zone to which the interfaces belong.

You can use these MIP addresses as the destination addresses in policies between any two zones, and as the source addresses when defining a policy from the Global zone to any other zone.

- **host** *ip_addr2* Specifies the IP address of a host device that uses IPv4 addressing. The **netmask** value specifies either a single one-to-one mapping or a mapping of one IP address range to another. (Note: Be careful to exclude the interface and gateway IP addresses, and any Virtual IP addresses in the subnet from the MIP address range.)
- **vrouter** *vrouter* Identifies the virtual router containing a route to the host device.
- **netmask** Specifies the range of host IP addresses.

Example: The following commands use a MIP to allow remote hosts to request HTTP services from a local HTTP server, located in a non-routable subnet, over a public WAN infrastructure. The MIP directly translates all outgoing source IP addresses into public addresses.

1. Set up ethernet interfaces.

```
unset interface ethernet2 ip
unset interface ethernet2 zone
unset interface ethernet3 ip
unset interface ethernet3 zone
```

```
set interface ethernet2 zone trust
set interface ethernet2 ip 10.100.2.1/24
set interface ethernet3 zone untrust
set interface ethernet3 ip 1.1.12.1/24
```

2. Create a MIP definition for the interface bound to the Untrust zone.

```
set interface ethernet3 mip 2.2.22.5 host 10.100.2.5 vrouter trust-vr
```

3. Create a policy definition that invokes the MIP.

```
set policy from untrust to trust any mip(2.2.22.5) http nat permit
save
```

mtu

```
set interface interface mtu number
unset interface interface mtu
```

mtu	Sets the Maximum Transmission Unit (MTU) for the interface. The MTU is the largest physical packet size (in octets), that the device can transmit on the interface. The NetScreen device must fragment any messages larger than the MTU before sending them. The default MTU size is 1500 octets. Enter a value between 800 and 1500.
------------	---

nat

set interface *interface* **nat**

nat Directs the device to perform Network Address Translation (NAT) on outbound traffic from the trusted LAN. This option is only available when the device is in Route Mode, in which the interfaces have assigned IP addresses.

nhtb

set interface *interface.number* **nhtb** *ip_addr* **vpn** *tunn_str*

unset interface *interface.number* **nhtb** *ip_addr*

nhtb Binds the specified VPN tunnel (**vpn** *tunn_str*) to the tunnel interface and manually maps the specified VPN tunnel to the IP address of a remote peer's tunnel interface (*ip_addr*) in the NHTB (next-hop tunnel binding) table. After that, you can enter a static route in the route table that uses that tunnel interface IP address as the gateway.

Example: With the following commands you first bind vpn1 to tunnel.1 and map vpn1 to 10.2.3.1, which is the IP address of the remote peer's tunnel interface. Then you define a static route to 10.2.2.0/24, which is the address of the remote peer's internal LAN, through tunnel.1 in the trust-vr routing domain, using the remote peer's tunnel interface IP address (10.2.3.1) as the next-hop gateway:

set interface **tunnel.1** **nhtb** **10.2.3.1** **vpn** **vpn1**

set vrouter **trust-vr** **route** **10.2.2.0/24** **interface** **tunnel.1** **gateway** **10.2.3.1**

phy

```
set interface interface phy { ... }
unset interface interface phy { ... }
```

phy

Defines the physical connection mode on the specified interface.

- **auto** The NetScreen unit automatically decides whether to operate at full or half duplex (as required by the network device connected to NetScreen unit).
- **full** Forces the NetScreen device to operate at full duplex. Specify either 100Mbps or 10Mbps.
- **half** Forces the NetScreen device to operate at half duplex. Specify either 100Mbps or 10Mbps.
- **holddown** *number* Sets the holddown time for the link, in increments of 100 milliseconds.
- **link-down** Forces the physical link down.
- **manual** Specifies manual mode for a gigabit interface. Setting the gigabit interface to **manual** disables auto negotiation. **Note:** You must configure both sides in the same negotiation mode, or the link does not initiate.

protocol

```
get interface interface protocol { ospf | rip }
set interface interface protocol ospf
{
  area { ip_addr | number } |
  authentication
  {
    active-md5-key-id id_num |
    md5 key_str [ key-id id_num ] |
    password pswd_str
  } |
  cost number |
  dead-interval number |
  enable |
  hello-interval number |
  link-type point-to-point |
```

```
neighbor-list number1 [ number2 [ number3 [ number4 ] ] ] |
passive |
priority number |
retransmit-interval number |
transit-delay number
}
set interface interface protocol rip
[
authentication
{
active-md5-key-id id_num |
md5 key_str [ key-id id_num ] |
password pswd_str
} |
enable |
metric number |
passive-mode |
route-map name_str |
split-horizon [ poison-reverse ]
]
unset interface interface protocol bgp
unset interface interface protocol ospf
{
area |
authentication { active-md5-key-id | md5 [ key-id id_num ] } |
cost |
dead-interval |
enable |
hello-interval |
link-type point-to-point |
neighbor-list |
passive |
```

```
    priority |
    retransmit-interval |
    transit-delay
  }
unset interface interface protocol rip
{
  authentication { active-md5-key-id | md5 [ key-id id_num ] } |
  enable |
  metric |
  passive-mode |
  route-map |
  split-horizon [ poison-reverse ]
}
```

protocol rip

Sets, unsets, or displays the current routing protocol settings for the interface.

- **route-map** *name_str* Specifies the route-map on which to filter incoming routes (routes learned by RIP) or outgoing routes (routes advertised by RIP).
 - **in** Specifies the route map is to be used for incoming routes.
 - **out** Specifies the route map is to be used for outgoing routes.
- **authentication** { **password** *pswd_str* | **md5** *key_str* **key-id** *id_num* } Specifies the authentication method used to verify RIP neighbors.
 - **password** specifies a clear-text password used for verification. If you specify password authentication, you must also specify an 8-byte password.
 - **md5** directs the NetScreen device to use the Message Digest version 5 (MD5) authentication algorithm for verification. If you specify MD5 authentication, you must also specify a 16-byte key and, optionally, a key identifier (the default identifier is 0). You can specify more than one MD5 key with different key identifier numbers (between 0-255). If there are multiple MD5 keys configured, you can use the **active-md5-key-id** option to select the key identifier of the key to be used for authentication.
- **enable** Enables RIP on the specified interface.
- **metric** *number* Configures the RIP metric for the specified interface. The default metric is 1.

- **passive-mode** Specifies that the interface is to receive but not transmit RIP packets.
- **split-horizon** Enables the split-horizon function on the specified interface. If **split-horizon** is enabled, RIP does not advertise routes learned from a neighbor back to the same neighbor. This avoids the routing-loop problem that occurs in some routing situations. If **split-horizon** is disabled, RIP advertises routes learned from a neighbor back to the same neighbor with a metric of 16. By default, **split-horizon** is enabled.

When you enable the **poison-reverse** switch, RIP still advertises routes learned from a neighbor back to the same neighbor, but defines the metric for those routes as infinity (16). This causes the neighbor to immediately remove the route, thus breaking a potential routing loop faster than with **split-horizon** alone. When you disable this switch, RIP advertises routes learned from a neighbor back to the same neighbor with the correct metric.

protocol ospf

Sets, unsets or displays the current routing protocol settings for the interface.

- **area** { *ip_addr* | *number* } Assigns the interface to the specified OSPF area. OSPF areas divide the internetwork into smaller, more manageable constituent pieces. This technique reduces the amount of information that each router must store and maintain about all the other routers.
- **authentication** { **md5** *key_str* [**key-id** *id_num*] | **password** *pswd_str* } Specifies the authentication method, including MD5 key string, the key identifier number (the default is 0), and password. You can specify more than one MD5 key with different key identifier numbers (between 0-255). If there are multiple MD5 keys configured, you can use the **active-md5-key-id** option to select the key identifier of the key to be used for authentication.
- **cost** *number* Specifies the desirability of the path associated with the interface. The lower the value of this metric, the more desirable the interface path.
- **dead-interval** *number* Specifies the maximum amount of time that the NetScreen device waits, after it stops receiving packets from the neighbor, before classifying the neighbor as offline.
- **disable** Disables OSPF on the interface, thus preventing transmission or receipt of OSPF packets through the interface.
- **hello-interval** *number* Specifies the amount of time in seconds that elapse between instances of the interface sending Hello packets to the network announcing the presence of the interface.

- **link-type** Configures the interface link type. By default, an ethernet interfaces is treated as an interface to a broadcast network with multiple attached routers. For broadcast networks, the Hello protocol elects a Designated Router and Backup Designated Router for the network.
 - **point-to-point** Configures the interface as a point-to-point link.
- **neighbor-list** *number1* [*number2* [*number3* [*number4*]]] Specifies the number of access lists (up to four), from which the local virtual router accepts valid neighbors to form adjacencies. The access list must be in the virtual router to which the interface is bound.
- **passive** Specifies that the IP address of the interface is advertised into the OSPF domain as an OSPF route and not as an external route, but the interface does not transmit or receive OSPF packets. This option is useful when BGP is also enabled on the interface.
- **priority** *number* Specifies the router election priority.
- **retransmit-interval** *number* Specifies the amount of time (in seconds) that elapses before the interface resends a packet to a neighbor that did not acknowledge a previous transmission attempt for the same packet.
- **transit-delay** *number* Specifies the amount of time (in seconds) that elapses before the NetScreen device advertises a packet received on the interface.

route

```
set interface interface route
```

route Directs the device to run in Route Mode, in which the interfaces have assigned IP addresses.

route-deny

```
set interface interface route-deny  
unset interface interface route-deny
```

route-deny Enabling this flag blocks all traffic in or out of the same interface. This includes traffic between the primary subnet and any secondary subnet, and one secondary subnet to another secondary subnet.

screen

get interface *interface* **screen**

screen Displays the current firewall (screen) counters.

secondary

set interface *interface* **ip** *ip_addr/mask* **secondary**

get interface *interface* **secondary** [*ip_addr*]

secondary Sets or displays the secondary address configured for the interface.

tag

set interface *interface.n* **tag** *id_num* **zone** *zone*

tag Specifies a VLAN tag (*id_num*) for a virtual (logical) subinterface. The interface name is *interface.n*, where *n* is an ID number that identifies the subinterface. For information on interface names, see ["Interface Names" on page A-1](#).

Example: The following command creates a subinterface for physical interface ethernet3/1, assigns it VLAN tag 300, and binds it to the Untrust zone:

set interface ethernet3/1.2 tag 300 zone untrust

track-ip

get interface *interface* **track-ip**

set interface *interface* **track-ip**

[
 dynamic |
 threshold *number* |
 ip *ip_addr*

```
    [
      interval number |
      threshold number |
      weight number
    ]
  ]
unset interface interface track-ip
[
  dynamic |
  ip ip_addr [ interval | threshold | weight ] |
  threshold
]
```

track-ip

Sets, unsets, or displays the tracking of IP addresses for the specified interface.

- **dynamic** Configures tracking of the IP address of the default gateway for the interface.
- **threshold number** Specifies the failure threshold for IP tracking on the interface. If the weighted sum of all tracked IP failures on the interface is equal to or greater than the threshold, IP tracking on the interface is considered to be failed and the routes associated with the interface are deactivated on the NetScreen device. On some NetScreen devices, failover to the backup interface occurs. Unsetting the tracked IP threshold on the interface sets the threshold to the default value of 1.
- **ip ip_addr** Configures tracking for the specified IP address. You can specify the following options:
 - **interval number** Specifies the interval, in seconds, that ping requests are sent to the tracked IP address. If you are unsetting the interval for the tracked IP address, the interval is changed to the default value of 1.
 - **threshold number** Specifies the failure threshold for the tracked IP address. If the number of consecutive ping failures to the tracked IP address is equal to or greater than the threshold, the tracked IP address is considered failed. If you are unsetting the threshold for the tracked IP address, the device changes the threshold to the default value (3).
 - **weight number** Specifies the weight associated with the failure of the tracked IP address. If a tracked IP address fails, its weight is used to calculate the weighted sum of all tracked IP failures on the interface. If you are unsetting the weight for the tracked IP address, the weight is changed to the default value of 1.

Examples: The following command defines IP tracking for an interface.

- IP address 1.1.1.1 on the ethernet3 interface
- Ping interval of 10 seconds
- Tracked IP address failure threshold of 5

```
set interface ethernet3 track-ip ip 1.1.1.1 interval 10 threshold 5
```

The following command sets the tracking threshold for the ethernet3 interface to 3:

```
set interface ethernet3 track-ip threshold 3
```

tunnel

```
set interface tunnel.n { zone name_str | protocol { bgp | ospf | rip } }
```

tunnel.n Specifies a tunnel interface. The *n* parameter is an ID number that identifies the tunnel interface.

Example: The following commands create a tunnel interface named **tunnel.2** with IP address **172.10.10.5/24**:

```
set interface tunnel.2 zone untrust
set interface tunnel.2 ip 172.10.10.5/24
```

vip

```
set interface interface vip ip_addr
[ + ] port_num [ name_str ip_addr [ manual ] ]
```

vip Defines a Virtual IP (VIP) address (*ip_addr*) for the interface so you can map routable IP addresses to internal servers and access their services. The *port_num* parameter is the port number, which specifies which service to access. The *name_str* and *ip_addr* parameters specify the service name and the IP address of the server providing the service, respectively. The **manual** switch turns off server auto detection. Using the **+** operator adds another service to the VIP.

Example: The following command creates a VIP for interface **ethernet3**, specifying the MAIL service (ID 25):

```
set interface ethernet3 vip 1.1.14.15 25 MAIL 10.1.10.10
```

vlan trunk

```
set interface vlan1 vlan trunk
unset interface vlan1 vlan trunk
```

vlan trunk (**vlan1** interface only.) Determines whether the NetScreen device accepts or drops Layer-2 frames. The device makes this decision only when the following conditions apply:

- The NetScreen device is in transparent mode.
- The device receives VLAN tagged frames on an interface.

The device then performs one of two actions.

- Drop the frames because they have tags.
- Ignore the tags and forward the frames according to MAC addresses.

The **vlan trunk interface** switch determines which action the device performs. For example, the command **set interface vlan1 vlan trunk** instructs the NetScreen device to ignore the tags and forward the frames. This action closely follows that of a Layer-2 switch "trunk port."

webauth

```
set interface interface webauth
```

webauth Enables WebAuth user authentication.

webauth-ip

```
set interface interface webauth-ip ip_addr
```

webauth-ip Specifies the WebAuth server IP address for user authentication. Before sending service requests (such as MAIL) through the interface, the user must first browse to the WebAuth address with a web browser. The NetScreen device presents a login screen, prompting for user name and password. After successfully entering the user name and password, the user can send service requests through the interface.

To protect an interface with the WebAuth feature, you must create a security policy with the **set policy** command, specifying the **webauth** switch. To specify the WebAuth server, use the **set webauth** command.

zone

```
set interface interface zone zone  
unset interface interface zone
```

zone Binds the interface to a security zone.

Example: To bind interface **ethernet2/2** to the Trust zone:

```
set interface ethernet2/2 zone trust
```

“ip” through “port-mode”

This section lists and describes Juniper Networks NetScreen Command Line Interface (CLI) commands **ip** through **port-mode**.

Note: As you execute CLI commands using the syntax descriptions in this section, you may find that certain commands and command features are unavailable on your NetScreen platform. A good example is the **vsys** command, which is available on a NetScreen-500 device, but not on a NetScreen-5XP device. Similarly, some command options are unavailable on certain platforms, as with the **df-bit** option of the **vpn** command. This option is available on a NetScreen-500, but not on a NetScreen-5XP.

Description: Use the **ip** commands to set or display IP parameters for communication with a TFTP server.

A NetScreen device can use TFTP servers to save or import external files. These files can contain configuration settings, software versions, public keys, error messages, certificates, and other items.

Syntax

get

```
get ip tftp
```

set

```
set ip tftp  
{  
  retry number |  
  timeout number  
}
```

Keywords and Variables

retry

```
set ip tftp retry number
```

retry	The number of times to retry a TFTP communication before the NetScreen device ends the attempt and generates an error message.
--------------	--

Example: The following command sets the number of retries to 7:

```
set ip tftp retry 7
```


timeout

```
set ip tftp timeout number
```

timeout Determines how the long (in seconds) the NetScreen device waits before terminating an inactive TFTP connection.

Example: The following command sets the timeout period to 15 seconds:

```
set ip tftp timeout 15
```

ip-classification

Description: Use the **ip-classification** command to display the current IP-based traffic classification.

IP-based traffic classification allows you to use virtual systems without VLANs. Instead of VLAN tags, the NetScreen device uses IP addresses to sort traffic, associating a subnet or range of IP addresses with a particular system (root or vsys).

Using IP-based traffic classification exclusively to sort traffic, all systems share the following:

- The untrust-vr and a user-defined internal-vr
- The Untrust zone and a user-defined internal zone
- An Untrust zone interface and a user-defined internal zone interface

To designate a subnet or range of IP addresses to the root system or to a previously created virtual system, you must issue one of the following CLI commands at the root level:

```
set zone zone ip-classification net ip_addr/mask { root | vsys name_str }  
set zone zone ip-classification range ip_addr1-ip_addr2 { root | vsys name_str }
```

For more information, see the [zone](#) command.

Syntax

get

```
get ip-classification [ zone zone ]
```

Keywords and Variables

zone

```
get ip-classification zone zone [ ip ip_addr ]
```

zone The name of the security zone. The **ip ip_addr** option specifies a particular address in a specific zone.

ippool

Definition: Use the **ippool** commands to associate the name of an IP pool with a range of IP addresses. The NetScreen device uses IP pools when it assigns addresses to dialup users using Layer 2 Tunneling Protocol (L2TP).

Syntax

get

```
get ippool name_str
```

set

```
set ippool string ip_addr1 ip_addr2
```

unset

```
unset ippool string
```

Keywords and Variables

Variable Parameters

```
get ippool name_str
```

```
set ippool string ip_addr1 ip_addr2
```

```
unset ippool string
```

<i>string</i>	Defines the name of the IP pool.
<i>ip_addr1</i>	Sets the starting IP address in the IP pool.
<i>ip_addr2</i>	Sets the ending IP address in the IP pool.

Example: To configure the IP pool named "office" with the IP addresses 172.16.10.100 through 172.16.10.200:

```
set ippool office 172.16.10.100 172.16.10.200
```

I2tp

Description: Use the **I2tp** commands to configure or remove L2TP (Layer 2 Tunneling Protocol) tunnels and L2TP settings from the NetScreen device.

L2TP is an extension to PPP (Point-to-Point Protocol) that allows ISPs to operate VPNs. L2TP allows dial-up users to make virtual Point-to-Point Protocol (PPP) connections to an L2TP network server (LNS). The NetScreen device can operate as such a server.

Syntax

clear

```
clear [ cluster ] l2tp { all | ip ip_addr }
```

get

```
get l2tp
{
    all [ active ] | tunn_str [ active ] |
    default
}
```

set (default)

```
set l2tp default
{
    auth server name_str [ query-config ] |
    ippool string |
    dns1 ip_addr | dns2 ip_addr |
    wins1 ip_addr | wins2 ip_addr |
    ppp-auth { any | chap | pap } |
}
```

set (tunn_str)

```
set l2tp tunn_str
[
  auth server name_str
    [ query-config ] [ user usr_name | user-group grp_name ] |
  [ peer-ip ip_addr ]
    [ host name_str ]
      [ outgoing-interface interface ]
        [ secret string ]
          [ keepalive number ] |
  remote-setting
    { [ ippool string ]
      [ dns1 ip_addr ]
        [ dns2 ip_addr ]
          [ wins1 ip_addr ]
            [ wins2 ip_addr ]
        }
  ]
```

unset

```
unset l2tp
{
  default { dns1 | dns2 | ippool | radius-port | wins1 | wins2 } |
  tunn_str
    {
      auth |
      host |
      keepalive |
      outgoing-interface interface { keepalive | secret } |
      peer-ip |
      remote-setting [ ippool ] [ dns1 ] [ dns2 ] [ wins1 ] [ wins2 ] |
      secret
    }
}
```

Keywords and Variables

Variable Parameter

```
get l2tp tunn_str
get l2tp tunn_str [ ... ]
set l2tp tunn_str [ ... ]
    unset l2tp tunn_str { ... }
```

tunn_str The name or IP address of the L2TP tunnel.

Example: The following command identifies the RADIUS authentication server (Rad_Serv) for an L2TP tunnel (Mkt_Tun).

```
set l2tp Mkt_Tun auth server Rad_Serv
```

active

```
get l2tp all active
get l2tp tunn_str active
```

active Displays the currently active L2TP connections for tunnels.

Example: The following command displays the current active/inactive status of the L2TP connection for a tunnel (home2work):

```
get l2tp home2work active
```

all

```
clear cluster l2tp all
clear l2tp all
get l2tp all
```

all Displays or clears the ID number, tunnel name, user, peer IP address, peer host name, L2TP tunnel shared secret, and keepalive value for every L2TP tunnel (**all**) or a specified L2TP tunnel (*string*).

auth server

```
set l2tp tunn_str auth server name_str [ ... ]
set l2tp default auth server name_str [ ... ]
unset l2tp tunn_str auth
```

auth server Specifies the object name (*name_str*) of the authentication server containing the authentication database.

- **query-config** Directs the NetScreen device to query the authentication server for IP, DNS, and WINS information.
- **user** *usr_name* Restricts the L2TP tunnel to a specified user (*usr_name*).
- **user-group** *grp_name* Restricts the L2TP tunnel to a specified user group (*grp_name*).

Example: The following command directs the device to query the RADIUS authentication server (Rad_Serv) for IP, DNS, and WINS information:

```
set l2tp Mkt_Tun auth server Rad_Serv query-config
```

cluster

```
clear cluster l2tp { ... }
```

cluster Propagates the **clear** operation to all other devices in a NSRP cluster.

default

```
get l2tp default
set l2tp default { ... }
unset l2tp tunn_str [ ... ]
unset l2tp default { ... }
```

default

Defines or displays the default L2TP settings.

- **auth server** *name_str* The object name of the authentication server.
- **dns1** *ip_addr* The IP address of the primary DNS server.
- **dns2** *ip_addr* The IP address of the secondary DNS server.
- **ippool** *string* The name of the L2TP IP pool, from which IP addresses are drawn to be assigned to L2TP users.
- **ppp-auth** { **any** [**chap** | **pap**] } Specifies the authentication type in response to a dialup user's request to make a Point-to-Point Protocol (PPP) link. (The **any** switch instructs the NetScreen device to negotiate CHAP and then, if that attempt fails, PAP.)
 - **chap** specifies Challenge Handshake Authentication Protocol (CHAP), which does not transmit the password across the network.
 - **pap** specifies Password Authentication Protocol (PAP), which does not use encryption.
- **radius-port** *port_num* Defines the port number of the default L2TP server. The number can be between 1024 and 65,535.
- **wins1** *ip_addr* The IP address of the primary WINS server.
- **wins2** *ip_addr* The IP address of the secondary WINS server.

Example: The following commands create a set of default L2TP settings.

- IP pool (chiba).
- Use of the local database.
- CHAP for PPP authentication.
- Primary and secondary DNS servers at 192.168.2.1 and 192.168.4.71 respectively.
- Primary and secondary WINS servers at 10.20.1.16 and 10.20.5.101 respectively.

```
set l2tp default ippool chiba
set l2tp default auth local
set l2tp default ppp-auth chap
set l2tp default dns1 192.168.2.1
set l2tp default dns2 192.168.4.71
set l2tp default wins1 10.20.1.16
set l2tp default wins2 10.20.5.101
```

host

```
set l2tp tunn_str [ ... ] host name_str [ ... ]
unset l2tp tunn_str host
```

host Adds a restriction that allows only a client with the specified client host name (*name_str*) to establish the L2TP tunnel.

keepalive

```
set l2tp tunn_str [ ... ] keepalive number
```

keepalive Defines how many seconds of inactivity, the NetScreen device (LNS) waits before sending a hello message to the dialup client (LAC).

Example: The following command specifies a keepalive value of 120 for an L2TP tunnel (west_coast):

```
set l2tp west_coast keepalive 120
```

outgoing-interface

```
set l2tp tunn_str [ ... ] outgoing-interface interface
```

outgoing-interface Specifies the outgoing interface for the L2TP tunnel. **Note:** This setting may be mandatory on your NetScreen device.

Example: The following command specifies interface **ethernet4** as the outgoing interface for L2TP tunnel (east_coast):

```
set l2tp east_coast outgoing-interface ethernet4
```

peer-ip

```
set l2tp tunn_str [ ... ] peer-ip ip_addr [ ... ]
```

peer-ip Adds a restriction that allows only a client host with the specified IP address (*ip_addr*) to establish the L2TP tunnel.

Example: The following command specifies the IP address of the LAC (172.16.100.19):

```
set l2tp east_coast peer-ip 172.16.100.19
```

secret

```
set l2tp tunn_str [ ... ] secret string [ ... ]
```

secret Defines a shared secret used for authentication between the NetScreen device (which acts as the L2TP Network Server, or LNS) and the L2TP access concentrator (LAC).

Example: The following command specifies a shared secret (94j9387):

```
set l2tp east_coast secret 94j9387
```

user

```
set l2tp tunn_str auth server name_str [ ... ] user usr_name
```

user Restricts the L2TP tunnel to a L2TP user (*usr_name*). (Not specifying *name_str* enables any L2TP user.)

Example: The following command adds a restriction that allows only a specified L2TP user (jking) to establish a L2TP tunnel (west_coast).

```
set l2tp west_coast auth server Our_Auth user jking
```

Defaults

The default L2TP UDP port number is 1701.

By default, the NetScreen device uses no L2TP tunnel secret to authenticate the LAC-LNS pair. This is not a problem, because the device performs IKE authentication when it uses L2TP over IPSec.

The default interval for sending a keepalive message is 60 seconds.

PPP-auth type is **any**.

led

Description: When either an event alarm or a firewall attack occurs, the LED glows red to signal the attack. Use the **clear led** command to return an ALARM or FW (firewall) LED to green after such an attack occurs.

Syntax

clear

```
clear [ cluster ] led { alarm | firewall }
```

Keywords and Variables

alarm

```
clear [ cluster ] led alarm
```

alarm Specifies the ALARM LED.

cluster

```
clear cluster led alarm  
clear cluster led firewall
```

cluster Propagates the **clear** operation to all other devices in a NSRP cluster.

firewall

```
clear [ cluster ] led firewall
```

firewall Specifies the firewall (FW) LED.

lcd

Description: Use the **lcd** commands to activate or inactivate the LCD on the front panel of a NetScreen device, or to display the current **lcd** setting.

Syntax

get

```
get lcd
```

set

```
set lcd  
  {  
    display |  
    key-in  
  }
```

unset

```
unset lcd  
  {  
    display |  
    key-in  
  }
```

Keywords and Variables

display

```
set lcd display  
unset lcd display
```

display Turns the LCD off or on and locks the control keys.

key-in

```
set lcd key-in  
unset lcd key-in
```

key-in Locks and unlocks the control keys, but does not affect the LCD display.

license-key

Description: Use the **license-key** command to upgrade or display the current software license.

The license key feature allows you to expand the capabilities of your NetScreen device without having to upgrade to a different device or system image. You can purchase a key that unlocks specified features already loaded in the software, such as the following:

- User capacity
- Virtual systems
- Zones
- Virtual routers
- HA

Syntax

exec

```
exec license-key
{
  nsrp key_str |
  update |
  vrouter key_str |
  vsys key_str |
  zone key_str
}
```

get

```
get license-key
```

set

```
set license-key update-url url_str
```

unset

```
unset license-key update-url
```

Keywords

nsrp

```
exec license-key nsrp key_str
```

nsrp Specifies a NetScreen Redundancy Protocol (NSRP) license key (*key_str*).

update

```
exec license-key update
```

update Before your NetScreen device can receive regular update service for Deep Inspection (DI) signatures, you must purchase a subscription to the service, register your device, and then retrieve the subscription. You retrieve the subscription and activate it on your device by executing the command **exec license-key update**. For more information, refer to the *NetScreen Concepts and Examples Reference Guide*.

update-url

```
set license-key update-url url_str
```

update-url Specifies the URL of the license key server from which the NetScreen device loads license key updates.

vrouter

exec license-key vrouter *key_str*

vrouter Specifies a virtual router license key (*key_str*).

vsys

exec license-key vsys *key_str*

vsys Specifies a virtual system (vsys) license key (*key_str*).

zone

exec license-key zone *key_str*

zone Specifies a security zone license key (*key_str*).

Description: Use the **log** commands to configure the NetScreen device for message logging.

The the **log** commands allow you to perform the following:

- Display the current log status according to severity level, policy, service, ScreenOS module, source, destination, or duration.
- Determine which log information to display or omit.
- Display asset recovery information.
- Configure logging to mitigate message loss due to memory limitations.

Syntax

clear

```
clear [ cluster ] log
{
  self [ end-time string ] |
  system [ saved ] |
  traffic [ policy id_num [ -id_num ] [ end-time string ] ]
}
```

get

```
get log
{
  asset-recovery |
  audit-loss-mitigation |
  self | traffic [ policy pol_num [ -pol_num ] ]
    [ start-date date [ time ] ] [ end-date date ] [ time ] ]
    [ start-time string ] [ end-time string ]
    [ min-duration string ] [ max-duration string ]
}
```

```
[ service name_str ]
  [ src-ip ip_addr [ -ip_addr ]
    [ src-netmask mask ] [ src-port port_num ]
  ]
    [ dst-ip ip_addr [ -ip_addr ] [ dst-netmask mask ] ]
    [ dst-port port_num ]
    [ no-rule-displayed ] |
sort-by
{
  date [ [ start-date date [ time ] ] [ end-date date [ time ] ] ]
  dst-ip [ ip_addr [ -ip_addr | dst-netmask mask ] ]
  src-ip [ ip_addr [ -ip_addr | src-netmask mask ] ]
  time
    [ start-time time ]
    [ end-time time ]
}
setting [ module { system | all } ]
}
```

set

```
set log
{
  audit-loss-mitigation |
  module name_str level string destination string
}
```

unset

```
unset log
{
  audit-loss-mitigation |
  module name_str level string destination string
}
```

Keywords and Variables

audit-loss-mitigation

```
get log audit-loss-mitigation
set log audit-loss-mitigation
unset log audit-loss-mitigation
```

audit-loss-mitigation Stops generation of auditable events when the number of such events exceeds the capacity of the NetScreen device. Enabling this feature reduces the loss of event logs due to log overloads. On some NetScreen devices, you must connect the syslog server to the management interface on the Management Module. This ensures that the syslog server is available if the audit trail fills up and network traffic stops.

cluster

```
clear cluster log { ... }
```

cluster Propagates the **clear** operation to all other devices in a NSRP cluster.

destination

```
set log module name_str level string destination string
unset log module name_str level string destination string
```

destination Specifies the destination of the generated log messages. The permissible destinations are **console**, **internal**, **e-mail**, **snmp**, **syslog**, **webtrends**, and **pcmcia**.

Example: The following command instructs the NetScreen device to direct all system module messages at the **alert** level (or higher) to the console port.

```
set log module system level alert destination console
```

dst-port

```
get log traffic dst-port number
```

dst-port Filters the output of the get log command by a range of destination port numbers or by a specific destination port number.

Example: The following command filters the get traffic log output to display only traffic destined for port 80 (that is, HTTP traffic):

```
get log traffic dst-port 80
```

level

```
set log module name_str level string destination string  
unset log module name_str level string destination string
```

level Specifies the minimum urgency level of the generated log messages. Starting with the most urgent, these levels are **emergency**, **alert**, **critical**, **error**, **warning**, **notification**, **information**, and **debugging**. For the **get log** command, the **all-levels** option displays all security levels.

Example: The following command instructs the NetScreen device to direct all system module messages at the **critical** level (or higher) to the e-mail server:

```
set log module system level critical destination email
```

min-duration | max-duration

```
get log event { ... } [ ... ] min-duration string [ ... ]
get log event { ... } [ ... ] max-duration string [ ... ]
```

min-duration	Displays traffic log entries for traffic whose duration was longer than or equal to the minimum duration specified.
max-duration	Displays traffic log entries for traffic whose duration was shorter than or equal to the maximum duration specified.

Example: The following command displays traffic log entries for traffic that lasted 5 minutes to 1 hour:

```
get log traffic min-duration 00:05:00 max-duration 01:00:00
```

module

```
get log event module { ... } [ ... ]
set log module name_str { ... }
unset log module name_str { ... }
```

module	Specifies the name of the ScreenOS module that generates the log message.
---------------	---

Example: The following command instructs the NetScreen device to direct all system module messages at the **critical** level (or higher) to the webtrends server:

```
set log module system level critical destination webtrends
```

no-rule-displayed

```
get log { ... } [ ... ] no-rule-displayed
```

no-rule-displayed	Displays traffic log entries, but does not display policy information.
--------------------------	--

Example: The following command displays traffic log entries without displaying policy information:

```
get log traffic no-rule-displayed
```

policy

```
clear [ cluster ] log traffic policy pol_num [ ... ]
```

policy Displays traffic log entries for a policy (specified by its ID number) or for several policies (specified by a range of ID numbers). The ID number can be any value between 0 and the total number of established policies. To define a range, enter the starting and ending ID numbers using this syntax: *pol_num - pol_num*

Example: The following command displays traffic log table entries for any policy with ID 3 to 9 (inclusive):

```
get log traffic policy 3-9
```

self

```
clear [ cluster ] log self [ ... ]  
get log self [ ... ]
```

self Clears or displays self-log entries from the log.

Example: The following command displays traffic log table entries for any policy with a source IP address of 172.16.10.1 and a destination address of 172.16.10.100:

```
get log self src-ip 172.16.10.1 dst-ip 172.16.10.100
```

service

```
get log { ... } [ ... ] service name_str [ ... ]
```

service Displays traffic log entries for a specified Service, such as **TCP**, **ICMP**, **FTP**, or **Any**. The name does not have to be complete; for example, both **TC** and **CP** are recognized as **TCP**. Although you cannot specify a Service group, note that because **TP** is recognized as **FTP**, **HTTP**, and **TFTP**, entering **TP** displays log entries for all three Services.

Example: The following command displays traffic log table entries for TCP:

```
get log self service tcp
```

setting

```
get log setting [ ... ]
```

setting Displays log setting information. The **module** *string* value specifies the name of the module for which the log settings apply.

Example: The following command displays traffic log settings for the system module:

```
get log setting module system
```

sort-by

```
get log { ... } sort-by date [ [ start-date date ] [ end-date date ] ] [ time ]
get log { ... } sort-by dst-ip [ ip_addr [ -ip_addr | dst-netmask mask ] ]
get log { ... } sort-by src-ip [ ip_addr [ -ip_addr | src-netmask mask ] ]
get log { ... } sort-by time [ start-time time ] [ end-time time ]
```

sort-by Sorts the log information by any of the following criterion.

- **date | time** Sorts the logs by date, time, or both.

The **start-date** option displays logs that occurred at or before the time specified. The **end-date** option displays logs that occurred at or after the time specified. The format for **start-date** and **end-date** *date* is *mm/dd[/yy-hh:mm:ss]*. The format for **start-time** and **end-time** is *hh:mm:ss*.

You can omit the year (the current year is the default), or express the year using the last two digits or all four digits. The hour, minute, and second are optional. The delimiter between the date and the time can be a dash or an underscore:

12/31/2002-23:59:00

12/31/2002_23:59:00

- **dst-ip** Sorts the traffic logs by destination IP address.
- **src-ip** Sorts the traffic logs by source IP address.

Example: The following command displays traffic log settings sorted by date and time:

```
get log traffic sort-by date start-date 11/21/2003-22:24:00
```

src-ip | dst-ip

```
get log { ... } [ ... ] src-ip ip_addr [ -ip_addr ] [ ... ]  
get log { ... } [ ... ] sort-by src-ip ip_addr [ -ip_addr ] [ ... ]  
get log { ... } [ ... ] dst-ip ip_addr [ -ip_addr ] [ ... ]  
get log { ... } [ ... ] sort-by dst-ip ip_addr [ -ip_addr ] [ ... ]
```

src-ip Displays traffic log entries for a specified source IP address or range of source IP addresses. Include the subnet mask for a source IP address to display traffic entries for all IP addresses in the same subnet as the specified source IP address. You cannot specify a source IP range and a source subnet mask simultaneously. You can also direct the device to sort event logs by source IP address.

dst-ip Displays traffic log entries for a specified destination IP address or range of destination IP addresses. You can specify the subnet mask for a destination IP address, but you cannot specify a destination IP range and destination subnet mask simultaneously. You can also direct the device to sort event logs by destination IP address.

Example: The following command displays traffic log entries for the range of destination IP addresses 172.16.20.5–172.16.20.200:

```
get log traffic dst-ip 172.16.20.5-172.16.20.200
```

src-port

```
get log { ... } [ ... ] src-port port_num [ ... ]
```

src-port Displays traffic log entries for a specified port number or range of source port numbers.

Example: The following command displays traffic log entries from the source port 8081:

```
get log traffic src-port 8081
```

start-date | end-date

```
get log { ... } [ start-date date [ time ] ] [ end-date date [ time ] ]  
get log { ... } sort-by date [ start-date date [ time ] ] [ end-date date [ time ] ]
```

start-date *date [time]* Specifies the lower and upper ends of a range of dates for traffic or self logs. You can omit the year (the current year is the default), or express the year using the last two digits or all four digits. The hour, minute, and second are optional. The delimiter between the date and the time can be a dash or an underscore:

12/31/2001-23:59:00

12/31/2001_23:59:00

start-time | end-time

```
get log { ... } [ start-time time ] [ end-time time ]  
get log { ... } sort-by [ start-time time ] [ end-time time ]
```

start-time *time* Specifies the lower and upper ends of a range of times for traffic or self logs. When you specify a
end-time *time* start-time and/or end-time, the device sorts or filters the logs based on the specified times, regardless of the date. Specify the time in the following format: **hh:mm:ss**.

Example: The following command displays event log entries from 3:00 P.M. on March 4, 2001 to 2:59:59 P.M. on March 6:

```
get log event start-time 03/04/01_15:00 end-time 03/06_14:59:59
```

system

```
clear [ cluster ] log system [ ... ]  
get log system [ reversely | saved ]
```

system Displays current system log information. The **saved** switch displays saved system log information. The **reversely** switch displays information in reverse order.

Example: The following command generates log messages generated from module **system**, and to generate only messages that are **critical** or greater:

```
set log module system level critical destination console
```

traffic

```
clear [ cluster ] log traffic [ ... ]  
get log traffic [ ... ]
```

traffic Specifies traffic log entries.

Example: The following command displays traffic log entries from the source port 8081:

```
get log traffic src-port 8081
```

mac

Description: Use the **mac** commands to configure a static Media Access Control (MAC) address for a NetScreen interface, or to display information on the current MAC configurations.

Note: You can only execute the *mac* commands when the device is configured in transparent mode.

Syntax

set

```
set mac mac_addr interface
```

get

```
get mac interface
```

unset

```
unset mac mac_addr
```

Keywords and Variables

Variable Parameters

<i>mac_addr</i>	Specifies the MAC address.
<i>interface</i>	Specifies the name of the interface, as with ethernet1 .

Example: The following command sets the MAC address on an NetScreen device to 111144446666 for the **ethernet7** interface:

```
set mac 111144446666 ethernet7
```

mac-learn

Description: Use the **mac-learn** commands to clear the entries in the Media Access Control (MAC) learning table, or to display information on the current MAC configurations.

This command functions only when an interface is in Transparent mode. When interfaces are in Transparent mode, the NetScreen device operates at Layer 2. The security zone interfaces do not have IP addresses, and the NetScreen device forwards traffic like a Layer 2 switch.

Syntax

clear

```
clear [ cluster ] mac-learn [ stats ]
```

get

```
get mac-learn [ interface ]
```

Keywords and Variables

Variable Parameter

```
get mac-learn interface
```

interface Identifies the interface.

cluster

```
clear cluster mac-learn [ ... ]
```

cluster Propagates the **clear** operation to all other devices in a NSRP cluster.

stats

clear [**cluster**] **mac-learn stats**

stats Clears the MAC learning table statistics.

memory

Description: Use the **memory** commands to set or display the memory allocation conditions.

Syntax

get

```
get memory [ id_num | all | cache | error | free | module id_num | used ]
```

Keywords and Variables

Variable Parameters

```
get memory id_num
```

id_num The task ID number.

all

```
get memory all
```

all Displays memory fragments.

cache

```
get memory cache
```

cache Displays malloc cache.

error

get memory error

error Displays erroneous memory fragments.

free

get memory free

free Displays free memory.

ipc

get memory ipc

ipc Displays memory statistics about inter-process communication (IPC).

kernel

get memory kernel [...]

kernel Displays memory statistics about the kernel heap.

module

get memory module *id_num*

module Displays a single memory module (*id_num*).

pool

`get memory pool`

pool Displays pooled memory.

used

`get memory used`

used Displays used memory.

mirror

Description: Use the **mirror** commands to mirror all traffic for at least one source interface to a destination interface. This command is useful for debugging and monitoring network traffic. For example, you can connect a sniffer to a destination interface to monitor traffic passing through multiple source interfaces.

Note: When a destination interface mirrors multiple source interfaces, the device may drop some frames due to a bandwidth mismatch.

Syntax

get

```
get mirror port
```

set

```
set mirror port source interface1 destination interface2
```

unset

```
unset mirror port source interface
```

Keywords and Variables

destination | *source*

```
set mirror port source interface1 destination interface2
```

destination Specifies the source and destination interfaces.

modem

Description: Use the **modem** commands to configure modem and dial-up settings for the serial link.

Syntax

get

```
get modem
[
  config |
  settings |
  state |
  stats
]
```

set

```
set modem
{
  idle-time number |
  interval number |
  isp name_str
  {
    account login string password pswd_str |
    primary-number string [ alternative-number string ] |
    priority number
  }
  retry number |
  settings name_str { active | init-strings string } |
  speed num
}
```

unset

```
unset modem
{
  idle-time number |
  interval number |
  isp name_str [ alternative-number ] |
  retry number |
  settings name_str |
  speed
}
```

exec

```
exec modem command string
```

Keywords and Variables

account

```
set modem isp name_str account login string password pswd_str
```

account Specifies the user login and password for the ISP account.

Example: The following command configures the login kgreen and the password bodie45 for the ISP account isp1:

```
set modem isp isp1 account login kgreen password bodie45
```

active

```
set modem settings name_str active
unset modem settings name_str
```

active Activates the specified modem settings and deactivates any other configured settings.

Example: The following command activates settings for the modem usr14400:

```
set modem settings usr14400 active
```

alternative-number

```
set modem isp name_str primary-number string alternative-number string
```

alternative-number Specifies an alternate phone number to access the ISP.

Example: The following command configures primary and alternate phone numbers to access the ISP 'isp1':

```
set modem isp isp1 primary-number 4085551212 alternative-number 4085551313
```

command

```
exec modem command string
```

command Sends Hayes AT commands to the modem.

config

```
get modem config
```

config Displays HDLC/PPP configuration.

idle-time

```
set modem idle-time number
unset modem idle-time number
```

idle-time Specifies the number of minutes that elapse with no traffic on the dial-up connection before the NetScreen device disconnects the modem. The default is 10 minutes. A value of 0 means the modem never disconnects, even if there is no traffic on the dial-up connection.

Example: The following command sets an idle time of 12 minutes:

```
set modem idle-time 12
```

init-strings

```
set modem settings name_str init-strings string
unset modem settings name_str
```

init-strings Specifies the initialization string for the specified modem.

Example: The following command sets an initialization string for the modem usr14400:

```
set modem settings usr14400 init-strings AT&FX4&A3&B1&D2&H1&I0&K1&M4&R2S7=60
```

interval

```
set modem interval number
unset modem interval number
```

interval Specifies the number of seconds between dial-up retries. The default is 60 seconds.

Example: The following command sets a dial-up interval of 45 seconds:

```
set modem interval 45
```

isp

```
set modem isp name_str { ... }  
unset modem isp name_str
```

isp Specifies the ISP .

Example: The following command configures the login kgreen and the password bodie45 for the ISP isp1:

```
set modem isp isp1 account login kgreen password bodie45
```

primary-number

```
set modem isp name_str primary-number string
```

primary-number Specifies the primary phone number to access the ISP. If your modem uses tone dial by default, but you want to use pulse dial, precede the phone number with a **P**. If your modem uses pulse dial by default, but you want to use tone dial, precede the phone number with a **T**.

Example: The following command configures the primary phone number to access the ISP isp1 and specifies tone dial:

```
set modem isp isp1 primary-number T4085551212
```

priority

```
set modem isp name_str priority number
```

priority Specifies the priority of this ISP for dial-up backup, relative to other ISPs that may be configured. A value of 1 is the highest priority.

Example: The following command configures the ISP isp1 as the highest priority for dial-up backup:

```
set modem isp isp1 priority 1
```

retry

```
set modem retry number
unset modem retry number
```

retry Specifies the number of times ScreenOS dials the primary number, and then the alternative-number, if the line is busy or there is no answer from the ISP. The default is 3 times.

Example: The following command sets the number of dial-up retries to 4:

```
set modem retry 4
```

settings

```
set modem settings name_str active | init-strings string
unset modem settings name_str
get modem settings
```

settings Configures settings for the specified modem.

Example: The following command activates settings for the modem usr14400:

```
set modem settings usr14400 active
```

speed

```
set modem speed number
unset modem speed
```

speed Specifies the maximum baud rate for the serial link between the device and the modem. The baud rate can be 9600, 19200, 38400, 57600, or 115200 bps. The default is 115200 bps.

Example: The following command sets a maximum baud rate of 56Kbps for the serial link:

```
set modem speed 57600
```


state

get modem state

state Shows modem state information.

stats

get modem stats

stats Shows modem status.

Description: Use the **nrt**p command to clear all NRTP (NetScreen Reliable Transfer Protocol) packet queues.

NRTP is for multicasting NSRP control messages to multiple receivers when NetScreen devices are in a redundancy cluster (interconnected through the High Availability ports). NRTP ensures that the master NetScreen device always forwards configuration and policy messages to the backup devices.

Syntax

clear

```
clear [ cluster ] nrt p queues
```

get

```
get nrt p
{
  counters ( all | receive number | send } |
  group |
  xmtq
}
```

Keywords and Variables

cluster

```
clear cluster nrt p queues
```

cluster Propagates the **clear** operation to all other devices in a NSRP cluster.

counters

```
get nrtip counters ( all | receive number | send )
```

- counters** Displays statistical information tracked by counters.
- **all** Displays all counter statistics.
 - **receive** [*number*] Displays only counter statistics for information that the device receives from other devices in the cluster. The optional *number* parameter is an ID number that identifies a particular device in the cluster.
 - **send** Displays only counter statistics for information that the device sends to other devices.

groups

```
get nrtip group
```

- group** Displays the ID numbers of devices belonging to the group, and a count of the devices in the group.

queues

```
clear [ cluster ] nrtip queues
```

- queues** Clears the NRTP packet queues.

xmtq

```
get nrtip xmtu
```

- xmtq** Displays the length of the queue containing packets awaiting ACK responses from other devices.

Description: Use the **nsmgmt** commands to set up a NetScreen device for configuration and monitoring by NetScreen-Security Manager 2004, an enterprise-level management application that configures multiple NetScreen devices from remote hosts.

The **nsmgmt** command can modify settings for two Security Manager components.

- The *management system*, a set of services that reside on an external host (called a *device server*). These services process, track, and store device management information.
- The *Agent*, a service that resides on the managed NetScreen device. The Agent receives configuration parameters from the management system and pushes it to ScreenOS. The Agent also monitors the device and transmits reports back to the management system.

Note: It is not advisable to use the **nsmgmt** command to initiate communication between the Management System and the Agent. How to perform this initiation depends upon the current availability of the NetScreen device.

If your NetScreen device has a known, accessible IP address, you can set initial configuration parameters from a client through the Security Manager UI.

You can also use the UI if your device receives IP addresses dynamically, as with DHCP or PPPoE.

If the device is new and does not have a known, accessible IP address, you can perform simple initial configuration by executing an encrypted configuration file called *Configlet*, which automatically executes several **nsmgmt** commands. Your network security administrator creates the *Configlet* file on the Security Manager GUI Interface and delivers it to your remote site, usually manually or via email.

For more information, refer to “Adding Devices” in NetScreen-Security Manager 2004 Administrator’s Guide.

Syntax

get

```
get nsmgmt
[
  proto-dist
  {
    table { bytes | packets } |
    user-service
  }
]
```

set

```
set nsmgmt
{
  enable |
  init
  {
    id string |
    installer name name_str password pswd_str |
    otp string
  } |
  report
  {
    alarm { attack | di | other | traffic } enable |
    log { config | info | self | traffic } enable |
    proto-dist
    {
      enable |
      user-service svc_name { ah | esp | gre | icmp | ospf | tcp | udp }
      { port_num1-port_num2 }
    } |
    statistics { attack | ethernet | flow | policy } enable
  }
}
```

```
    }  
    server  
        primary | secondary  
            { name_str | ip_addr } [ port number ] [ src-interface interface ] |  
    }
```

unset

```
unset nsmgmt  
{  
    enable |  
    init { id | installer | otp } |  
    report  
        {  
            alarm { attack | di | other | traffic } enable |  
            log { config | info | self | traffic } enable |  
            proto-dist  
                {  
                    enable |  
                    user-service svc_name { ah | esp | gre | icmp | ospf | tcp | udp }  
                } |  
            statistics { attack | ethernet | flow | policy } enable  
        }  
    server { primary | secondary { name_str | ip_addr } }  
}
```

Keywords and Variables

enable

```
get nsmgmt enable
set nsmgmt enable
```

enable Enables remote management by initiating contact with the management server.

init

```
get nsmgmt init
set nsmgmt init id string
set nsmgmt init installer name name_str password pswd_str
set nsmgmt init otp string
```

init Sets initialization parameters for interaction with the management server.

- **id *string*** An ID used (only once) during initiation of the connection between the NetScreen device and the management server. The NetScreen device passes the ID to the Management System to look up the One-Time Password in the management database.
- **installer name *name_str* password *pswd_str*** Specifies an installer name and password, used (only once) during initiation of the connection between the NetScreen device and the management server.
- **otp *string*** Sets the One-Time Password (OTP). The NetScreen device uses this password one time to contact the Security Management system. After initiation of contact between the device and the management database, the device executes an **unset** command to erase the OTP.

keys

```
get nsmgmt keys
```

keys Displays information about the public and private keys used to encrypt and decrypt the Configlet file.

report

```
set nsmgmt report { alarm | log | proto-dist | statistics } { ... }  
unset nsmgmt report { alarm | log | proto-dist | statistics } { ... }
```

report

Specifies which event messages the NetScreen device transmits to the server.

alarm Enables the transmission of alarm events. The categories of alarms are as follows.

- **attack** Transmits attack alarms such as syn-flag or syn-flood. For more information on such attacks, see ["interface" on page 419](#).
- **di** Transmits attack alarms generated during Deep Inspection.
- **traffic** Transmits traffic alarms.
- **other** Transmits alarms other than attack, Deep Inspection, or traffic alarms.

The **enable** switch enables messaging for the specified alarm message.

log Enables the transmission of log events. The categories of logs are as follows.

- **config** Transmits log messages for events triggered by changes in device configuration.
- **info** Transmits low-level notification log messages about non-severe changes that occur on the device, as when an authentication procedure fails.
- **self** Transmits log messages concerning dropped packets (such as those denied by a policy) and traffic that terminates at the NetScreen device (such as administrative traffic). The self log displays the date, time, source address/port, destination address/port, duration, and service for each dropped packet or session terminating at the NetScreen device.

proto-dist Sets or displays parameters for transmission of messages concerning protocol distribution parameters.

- **enable** Enables transmission of protocol distribution messages to the server.
- **table** Displays the number of bytes or packets transmitted to the protocol distribution table.
- **user-service** *svc_name* Specifies messages generated by the following services.
 - **ah** AH (Authentication Header) service.
 - **esp** ESP (Encapsulating Security Payload) service.

- **gre** GRE (Generic Routing Encapsulation).
- **icmp** ICMP (Internet Control Message Protocol).
- **ospf** OSPF (Open Shortest Path First).
- **tcp** TCP (Transmission Control Protocol).
- **udp** UDP (User Datagram Protocol).

The *port_num1-port_num2* setting specifies a range of port numbers.

- **traffic** Transmits alarms generated while the device monitors and records the traffic permitted by policies. A traffic log notes the following elements for each session:
 - Date and time that the connection started
 - Source address and port number
 - Translated source address and port number
 - Destination address and port number
 - The duration of the session
 - The service used in the session

The **enable** switch enables messaging for the specified log message.

statistics Enables the the NetScreen device for reporting statistical information to the server.

- **attack** Enables transmission of messages containing attack statistics.
- **ethernet** Enables transmission of messages containing ethernet statistics.
- **flow** Enables transmission of messages containing traffic flow statistics.
- **policy** Enables transmission of messages containing policy statistics.

The **enable** switch enables messaging for the specified statistical message.

server

```
get nsmgmt server
set nsmgmt server { primary | secondary } { name_str | ip_addr } [ port number ]
```

server Identifies the Security Management system server.

Description: Use the **nsrp** commands to assign a NetScreen security device to a failover cluster, and to create and configure a Virtual Security Device (VSD) group for the cluster.

The purpose of a VSD group is to allow failover between two or more NetScreen security devices within a defined cluster. Each VSD group represents a group of devices in a cluster, elects a master device from the cluster, and provides a virtual security interface (VSI) that external devices use to reference the devices in the cluster.

A group may contain every device in the cluster. For example, if you give three devices the same cluster id, you can create a VSD group containing all three devices. A device can be in more than one VSD group at a time. For example, a device can be a master in one VSD group, while serving as a backup in another.

The basic steps needed to set up failover VSD groups are as follows.

1. Set up a cluster of devices using the **set nsrp cluster** command. This command assigns an identical cluster id to each device.
2. Set up a VSD group for the cluster using the **set nsrp vsd-group** command.
3. Set up a virtual security interface (VSI) for the VSD group using the **set interface** command.

Syntax

clear

```
clear [ cluster ] nsrp counter
    [
    packet-fwk |
    protocol |
    rto
    ]
```

exec

```
exec nsrp
{
  probe interface [ mac_addr ] [ count number ] |
  sync
  {
    file [ name filename ] from peer |
    global-config [ check-sum | save ] |
    rto
    {
      all |
      arp |
      auth-table |
      dhcp |
      dns |
      l2tp |
      phase1-SA |
      pki |
      rm |
      session |
      vpn
    }
    { from peer }
  } |
  vsd-group grp_num mode { backup | ineligible | init | pb }
}
```

get

```
get nsrp
[
  cluster |
  counter [ packet-fwd | protocol | rto ] |
  group |
```

```
ha-link |
monitor [ all | interface | track-ip | zone ] |
rto-mirror |
track-ip [ ip ip_addr ] |
vsd-group [ id id_num | all ]
]
```

set

```
set nsrp
{
  active-active transparent |
  arp number |
  auth password pswd_str |
  cluster [ id number | name name_str ] |
  config sync |
  data-forwarding |
  encrypt password pswd_str |
  ha-link probe [ interval number ] [ threshold number ]
  interface interface |
  link-hold-time number |
  link-up-on-backup |
  [ vsd-group id id_num ] monitor
  {
    interface interface [ weight number ] |
    threshold number |
    track-ip
    {
      ip
      [ ip_addr
      [
        interface interface |
        interval number |
        method { arp | ping } |
        threshold number |

```

```
        weight number
      ]
    ]
    threshold number |
    weight number |
  } |
  zone zone [ weight number ]
} |
rto-mirror
{
  hb-interval number |
  hb-threshold number |
  id id_num { direction { in | out } } |
  session [ off | non-vsi ] |
  sync
}
secondary-path interface |
track-ip
[
  ip
  [ ip_addr
    [
      interface interface |
      interval number |
      method { arp | ping } |
      threshold number |
      weight number
    ]
  ]
  threshold number
] |
vsd-group
{
  id id_num
  [
    mode ineligible |
```

```
        preempt [ hold-down number ] |
        priority number
    ] |
    hb-interval number |
    hb-threshold number |
    init-hold number |
    master-always-exist
}
}
```

unset

```
unset nsrp
{
    active-active transparent |
    arp number |
    auth |
    cluster id |
    config sync |
    data-forwarding |
    encrypt |
    ha-link probe [ interval ] [ threshold ] |
    link-hold-time |
    link-up-on-backup |
    monitor
    [
        interface interface [ weight ] |
        threshold |
        track-ip weight |
        zone zone [ weight ]
    ]
    [ all ] |
    rto-mirror
    {
        hb-interval number |
```

```
    hb-threshold number |
    id id_num { direction { in | out } } |
    session [ off | non-vsi ] |
    sync
    master-always-exist
  }
secondary-path |
track-ip
{
  ip [ ip_addr ]
  [
    interface |
    interval number |
    method { arp | ping } |
    threshold number |
    weight number
  ]
}
vsd-group
[
  all |
  id number [ mode | preempt | priority ] |
  hb-interval number |
  hb-threshold number |
  init-hold number
]
```

Keywords and Variables

active-active transparent

```
set nsrp active-active transparent
unset nsrp active-active transparent
```

active-active transparent Enables or disables transparent active-active mode.

arp

```
set nsrp arp number
unset nsrp arp number
```

arp Sets the number of gratuitous ARP requests that a newly elected master unit sends out, notifying other network devices of its presence. The default is 4.

Example: The following command instructs the NetScreen device to send out seven ARP requests:

```
set nsrp arp 7
```

auth

```
set nsrp auth password pswd_str
unset nsrp auth
```

auth Instructs the NetScreen device to authenticate NSRP communications using the specified password. Valid passwords contain from 1 to 15 characters.

Example: The following command sets the NSRP authentication password to "swordfish":

```
set nsrp auth password swordfish
```


cluster

```
get nsrp cluster
set nsrp cluster id number
```

cluster id Assigns the NetScreen device to a cluster, expressed as an integer (from 1 to 7, inclusive) to identify the cluster.

Example: The following command assigns the NetScreen device to cluster 2:

```
set nsrp cluster id 2
```

cluster (clear)

```
clear cluster nsrp counter [ ... ]
```

cluster Propagates the **clear** operation to all other devices in a NSRP cluster.

config sync

```
set nsrp config sync
unset nsrp config sync
```

config sync Enables or disables synchronization of device configuration between members of the NSRP cluster. After you enable this setting, any configuration change automatically propagates to the other devices in the cluster.

counter

```
clear [ cluster ] nsrp counter [ packet-fwd | protocol | rto ]
get nsrp counter [ packet-fwd | protocol | rto ]
```

- counter** Clears or displays the NSRP counter values.
- **packet-fwd** Clears or displays packet-forwarding counters only.
 - **protocol** Clears or displays NSRP protocol counters only.
 - **rto** Clears or displays RTO message counters only.

data-forwarding

```
set nsrp data-forwarding
unset nsrp data-forwarding
```

- data-forwarding** Enables or disables packet forwarding. The default setting is enabled.

encrypt password

```
set nsrp encrypt password pswd_str
unset nsrp encrypt
```

- encrypt password** Specifies that NSRP communications be encrypted using the specified password. Valid passwords contain from 1 to 15 characters.

Example: The following command sets the NSRP encryption password to "manta":

```
set nsrp encrypt password manta
```

group

get nsrp group

group Displays information on the VSD group.

ha-link probe

set nsrp ha-link probe [*interval number*] [*threshold number*]
unset nsrp ha-link probe [*interval*] [*threshold*]

ha-link probe Specifies the automatic sending of NSRP probe requests on all interfaces that are bound to the HA zone. If a reply is received from the peer within the threshold, the HA link is considered to be up. If the number of consecutive probe requests sent without receiving a reply from the peer reaches or exceeds the threshold, the HA link is considered to be down. You can specify the following optional parameters:

- **interval number** Specifies the interval, in seconds, at which probe requests are sent. Enter a number between 0 and 255. If you do not specify an interval, probe requests are sent every second.
- **threshold number** Specifies the failure threshold for the HA link. If the number of consecutive probe requests sent without receiving a reply from the peer reaches or exceeds the threshold, the HA link is considered to be down. Enter a value between 0 and 255. The default threshold is 5.

interface

set nsrp interface *interface*

interface The name of the interface to serve as the high-availability port. For information on interfaces, see ["Interface Names" on page A-1](#).

Example: The following command specifies that the NSRP interface is ethernet4:

set nsrp interface ethernet4

link

```
get nsrp link
```

link Displays HA link information

link-hold-time

```
set nsrp link-hold-time number  
unset nsrp link-hold-time
```

link-hold-time The delay time (in seconds) before the NetScreen device brings up the link with the peer device.

link-up-on-backup

```
set nsrp link-up-on-backup  
unset nsrp link-up-on-backup
```

link-up-on-backup Specifies that the link is always up on the backup device.

monitor

```
get nsrp monitor [ zone | interface | track-ip ] [ all ]  
set nsrp [ vsd-group id id_num ] monitor { ... }  
unset nsrp [ vsd-group id id_num ] monitor { ... }
```

monitor Specifies monitoring of NSRP objects (a physical interface, a zone, or tracked IP addresses) to determine VSD or device failure. You can specify the following parameters:

- **vsd-group id** *id-num* Identifies the Virtual Security Device (VSD) to which the threshold or monitored objects you configure applies. If you do not specify a VSD, the threshold or monitored objects you configure apply to the entire device.
- **all** Displays monitoring information for the device and all VSDs. If you specify **vsd-group id**, only monitoring information for the VSD is displayed.

- **interface** *interface* [**weight** *number*] Identifies the interface to be monitored and the weight that failure of the interface contributes to the failover threshold. The default weight is 255.
- **threshold** *number* Specifies a failover threshold which determines the failure of a specific VSD (if a VSD is specified) or failure of the device (if no VSD is specified). If the cumulative weight of the failure of all monitored objects (a physical interface, a zone, or tracked IP addresses) exceeds the threshold, the VSD or the device fails. The default threshold value is 255.
- **track-ip** **weight** *number* [**threshold** *number*] [**ip** *ip_addr*] Enables tracked IP object monitoring and the weight that failure of the tracked IP object (all tracked IP addresses) contributes to the device or VSD failover threshold. The default **weight** value is 255. The **threshold** value is the total weight of failed tracked IP addresses that determines failure of the tracked IP object. The default **threshold** value is 255. Specifies monitoring of tracked IP addresses to determine VSD or device failure. For each ip *ip_addr*, you can configure the following:
 - **interface** *interface* Specifies the outgoing interface through which the NetScreen device performs tracking. for the specified IP address. If you do not specify an interface, ping tries to find an outgoing interface from routing table entries and arp tries to find an outgoing interface within the same subnet. If an interface is not found, the tracking attempt fails.
 - **interval** *number* Specifies the interval, in seconds, between ping or ARP attempts to the specified IP address. Enter a vlaue between 1 and 200. The default is 1.
 - **method** { **arp** | **ping** } Specifies the method used for tracking the specified IP address. The default is ping.
 - **threshold** *number* Defines the number of failed tracking attempts that can occur before the tracking of the specified IP address is considered failed. The default is 3.
 - **weight** *number* Defines the weight of the failed tracking of the specified IP address. The default is 1.
- **zone** *zone* [**weight** *number*] Identifies the zone to be monitored and the weight that failure of all physical interfaces in the zone contributes to the failover threshold. The default weight is 255.

probe

```
exec nsrp probe interface [ mac_addr ] [ count number ]
```

- probe** Directs the device to immediately begin sending an NSRP probe request every second on an HA zone interface, for the number of times specified by **count**. If the peer receives a reply, the HA link is considered to be up. (If the request times out before the peer receives a reply, the HA link is considered to be down.) The device takes no action if there is no reply. (See ["ha-link probe" on page 301.](#))
- *interface* Identifies the HA zone interface on which probe requests are sent. You must specify an interface that is bound to the HA zone.
 - *mac_addr* Identifies the destination MAC address of an HA interface on a peer device. If you do not specify a destination MAC address, the device uses the default NSRP MAC address to send the probe request.
 - **count number** Specifies the number of times that the device sends the probe request. Enter a number greater than or equal to 1. (The default is 1.)

rto-mirror

```
get nsrp rto-mirror
set nsrp rto-mirror { ... }
unset nsrp rto-mirror { ... }
```

- rto-mirror** Creates an optional RTO mirror between two devices in a VSD group to back up runtime objects (RTOs).
In most cases, using this option is not necessary. Normally, RTO objects synchronize after execution of the **set nsrp rto sync** command.
A NetScreen device can belong to only one RTO mirror group at a time.
- **id id_num** Identifies the VSD group using its identification number *id_num*, an integer value between 1 and 127 inclusive. The **direction** setting determines whether the RTO mirror group direction is inbound or outbound.
 - **hb-interval number** Specifies the heartbeat interval in seconds.

- **hb-threshold** *number* Specifies the heartbeat-lost threshold. The minimum threshold value is 16 heartbeats.
- **session off** Disables the RTO session.
- **session non-vsi** Enables non-vsi session synchronization.
- **sync** Enables RTO object synchronization.

Example: The following command specifies that the RTO mirror group (10) direction is inbound:

```
set nsrp rto-mirror id 10 direction in
```

secondary-path

```
set nsrp secondary-path interface
```

```
unset nsrp secondary-path
```

secondary-path Specifies a secondary NSRP link interface.

Example: The following command specifies that the secondary NSRP link interface is ethernet5:

```
set nsrp secondary-path ethernet5
```

sync

```
exec nsrp sync { ... }
```

sync Specifies the name of a particular configuration, file, or RTO to copy from one unit to the other.

- **file** Specifies synchronization of the files in flash memory.
 - **name** *filename* specifies a particular file in flash memory. (Executing the **file** option without specifying a file name copies all the files.)
 - **from peer** specifies all files from the peer device.
- **global-config** Specifies synchronization of the current device configurations. The **check-sum** switch compares the check-sum after synchronization. The **save** switch synchronizes the PKI configuration and saves the synchronization configuration to flash memory.

- **rto** Specifies synchronization of the current runtime objects (RTOs) in the RTO mirror.
 - **all** Specifies all possible real-time objects.
 - **arp** Specifies the ARP (Address Resolution Protocol) information.
 - **auth-table** Specifies the authentication table information.
 - **dhcp** Specifies DHCP (Dynamic Host Configuration Protocol) information.
 - **dns** Specifies the DNS (Domain Name Service) information.
 - **pki** Specifies certificate information.
 - **phase1-sa** Specifies information on IKE Phase-1 SAs (Security Associations)
 - **rm** Specifies Resource Manager information.
 - **session** Specifies the session information.
 - **vpn** Specifies all VPN (Virtual Private Network) information.

Example: The following command instructs the NetScreen device to synchronize all runtime objects:

```
exec nsrp sync rto all from peer
```

track-ip

```
get nsrp track-ip [ ip ip_addr ]  
set nsrp track-ip [ ... ]  
unset nsrp track-ip [ ... ]
```

track-ip Enables path tracking, which is a means for checking the network connection between a NetScreen interface and that of another device. The IP address *ip_addr* identifies the other network device to check.

Executing **unset nsrp track ip** resets the track options to their default values.

ip *ip_addr*

- **interface** *interface* Specifies the interface through which the NetScreen device performs the path tracking. If you do not specify an interface, the device automatically chooses the interface for IP tracking using either the ping or arp method. If ping is used, the device tries to find an outgoing interface from entries in the routing table. If arp is used, the device tries to find an outgoing interface within the same subnet. If an interface is not found, the tracking attempt fails.
- **interval** *number* Specifies the interval in seconds between path tracking attempts. Required value is between 1 and 200. The default is 1.
- **method { arp | ping }** Specifies the method used for path tracking. The default is **ping**.
- **threshold** *number* Defines the number of failed tracking attempts that can occur before the tracking of the IP address is considered failed. The default is 3.
- **weight** *number* Defines the path weight. Valid weights are between 1 and 255 inclusive. The default weight is 1.

threshold *number*

Defines the sum of the weights of the tracked IP addresses that determine failover. The default is 255.

Example: The following command enables path tracking through interface ethernet4 to a device at IP address 172.16.10.10:

```
set nsrp track-ip ip 172.16.10.10 interface ethernet4
```

vsd-group

```
get nsrp vsd-group [ id id_num | all ]
set nsrp vsd-group [ ... ]
unset nsrp vsd-group [ ... ]
```

vsd-group

Configures a VSD group for a cluster.

id *id_num*

Creates a VSD group, identified by *id_num* (from 1 to 8, inclusive), that contains all members belonging to a single cluster of devices. Once created, a VSD group elects a master unit from the cluster it contains. Other devices reference the device cluster in the VSD group through the group's virtual security identification (VSI).

- **mode ineligible** Determines the running mode of the security device. The **ineligible** switch specifies that the local device is not intended for failover, even after system restart. (This may be necessary for administrative reasons.) Executing **unset nsrp vsd-group id number mode ineligible** specifies that the device is eligible again.
- **preempt [hold-down number]** Determines if the master unit keeps its master status until the unit itself relinquishes that status. To prevent rapid failovers, the master device waits for the specified hold-down interval, expressed as a number between 0 to 600 seconds inclusive. The default is 3.
- **priority number** The priority level of the device, expressed as an integer from 1 to 254, inclusive. The priority level determines the failover order for the device. The failover order determines which unit is the master unit when two NetScreen devices in a redundant group power up simultaneously, and which backup unit becomes the next master during a failover. (The unit with the number closest to 1 becomes the master unit.)

init-hold

The number of heartbeats that occur before the system exits the initial state (init mode). This value can be an integer from 5 to 255. The default is 5.

hb-interval number

Specifies the heartbeat interval, expressed in milliseconds. This value can be an integer from 200 to 1000. The default is 1000.

hb-threshold number

Specifies the heartbeat-lost threshold, the number of lost heartbeats allowed before failure. This value can be an integer from 3 to 255. The default is 3.

master-always-exist

Directs the system to elect a master and keep it operative even if all units in the NSRP cluster fail (by monitoring result). For example, if you disable **master-always-exist**, and two units tracking an IP later fail due to monitoring results, both units become inoperable and traffic cannot go through. If you enable **master-always-exist**, and both units fail, the cluster still elects a master unit, which remains operable, thus allowing traffic through.

Examples: The following command disables the local device for failover:

```
set nsrp vsd-group id 2 mode ineligible
```

The following command specifies that ten heartbeats must occur before the device exits the initial state:

```
set nsrp vsd-group init-hold 10
```

vsd-group (exec)

```
exec nsrp vsd-group grp_num mode { ... }
```

- | | |
|---------------------|--|
| vsd-group | Specifies a VSD group and the NetScreen device's new mode. |
| grp_num mode | <ul style="list-style-type: none">• In backup mode, the device takes over work for the master device when the master device fails.• In ineligible mode, the device is unavailable as a backup for the master device.• In init mode, the device is in the transient state that occurs when it joins the VSD group. (At the end of this initial hold up time, the device transitions to another state, such as master, backup, or primary backup.)• In pb (primary backup) mode, the unit is the first to take over when the master unit fails. |

Example: The following command instructs the NetScreen device to take over when the master unit fails:

```
exec nsrp vsd-group 2 mode pb
```

Defaults

The default value of **preempt [holddown number]** is zero.

The default value of **vsd-group id id_num priority number** is 100.

The default value of **vsd-group id id_num hb-interval number** is **1000** (1,000 milliseconds, or one second).

Creating an NSRP Cluster

The following commands set up an NSRP cluster consisting of two NetScreen devices

- Two VSD groups for the cluster
- VSI for the VSD group
- RTO object synchronization enabled, including session synchronization

On Device A

Trust Zone Redundant Interface and Manage IP

```
set interface redundant2 zone trust
set interface ethernet2/1 group redundant2
set interface ethernet2/2 group redundant2
set interface redundant2 manage-ip 10.1.1.3
```

Cluster and VSD Groups

```
set nsrp cluster id 1
set nsrp vsd-group id 0 preempt hold-down 10
set nsrp vsd-group id 0 preempt
set nsrp vsd-group id 0 priority 1
set nsrp vsd-group id 1
set nsrp monitor interface redundant2
set nsrp rto-mirror sync
save
```

On Device B

Trust Zone Redundant Interface and Manage IP

```
set interface redundant2 zone trust
set interface ethernet2/1 group redundant2
set interface ethernet2/2 group redundant2
```

```
set interface redundant2 manage-ip 10.1.1.4
```

Cluster and VSD Groups

```
set nsrp cluster id 1
set nsrp rto-mirror sync
set nsrp vsd-group id 1 priority 1
set nsrp vsd-group id 1 preempt hold-down 10
set nsrp vsd-group id 1 preempt
set nsrp monitor interface redundant2
set nsrp arp 4
set arp always-on-dest
```

Untrust Zone Redundant Interface

```
set interface redundant1 zone untrust
set interface ethernet1/1 group redundant1
set interface ethernet1/2 group redundant1
```

Virtual Security Interfaces

```
set interface redundant1 ip 210.1.1.1/24
set interface redundant2 ip 10.1.1.1/24
set interface redundant1:1 ip 210.1.1.2/24
set interface redundant2:1 ip 10.1.1.2/24
```

Routes

```
set vrouter untrust-vr route 0.0.0.0/0 interface redundant1 gateway 210.1.1.250
set vrouter untrust-vr route 0.0.0.0/0 interface redundant1:1 gateway 210.1.1.250
save
```

Description: Use the **ntp** commands to configure the NetScreen device for Simple Network Time Protocol (SNTP). SNTP is a simplified version of NTP, which is a protocol used for synchronizing computer clocks in the Internet. This version is adequate for devices that do not require a high level of synchronization and accuracy. To enable the SNTP feature, use the **set clock ntp** command.

Syntax

exec

```
exec ntp [ server { backup1 | backup2 | primary } ] update
```

get

```
get ntp
```

set

```
set ntp
{
  auth { preferred | required } |
  interval number |
  max-adjustment number |
  no-ha-sync |
  server [ { backup1 | backup2 } { ip_addr | dom_name } ]
    key-id number preshare-key string |
  timezone number1 number2
}
```

unset

```
unset ntp
{
  auth |
  interval |
  max-adjustment |
  no-ha-sync |
  server [ backup1 | backup2 ] [ key-id ] |
  interval |
  timezone
}
```

Keywords and Variables

auth

```
set ntp auth { preferred | required }
```

auth

Configures an authentication mode to secure NTP traffic between the NetScreen device and the NTP server.

- **required** The Required mode specifies that the NetScreen device must authenticate all NTP packets using the key id and preshared key information that the NetScreen device and the NTP server previously exchanged out of band (the device does not exchange the preshared key over the network).
- **preferred** The Preferred mode specifies that the NetScreen device first must try to authenticate all NTP packets by sending out an update request that includes authentication information—key id and checksum—same as for the Required mode. If authentication fails, the NetScreen device then sends out another update request without the authentication information.

Note: Before you can set an authentication mode, you must assign a key id and preshared key to at least one of the NTP servers configured on the NetScreen device.

interval

```
set ntp interval number  
unset ntp interval
```

interval Defines in minutes how often the NetScreen device updates its clock time by synchronizing with the NTP server. The range for the synchronization interval is from 1 to 1440 minutes (24 hours).

Example: The following command configures the NetScreen device to synchronize its clock time every 20 minutes:

```
set ntp interval 20
```

max-adjustment

```
set ntp max-adjustment number  
unset ntp max-adjustment
```

max-adjustment Configures a maximum time adjustment value. This value represents the maximum acceptable time difference between the NetScreen device system clock and the time received from a NTP server. When receiving a reply from a NTP server, the NetScreen device calculates the time difference between its system clock and the NTP server and updates its clock only if the time difference between the two is within the maximum time adjustment value that you set.

no-ha-sync

```
set ntp no-ha-sync  
unset ntp no-ha-sync
```

no-ha-sync In a high-availability configuration, instructs the NetScreen device not to synchronize its peer device with the NTP time update.

server

```
set ntp server { ip_addr | dom_name }
set ntp server key-id number preshare-key string
set ntp server { backup1 | backup2 } { ip_addr | dom_name }
set ntp server { backup1 | backup2 } key-id number preshare-key string
unset ntp server { ... }
```

- server**
- *ip_addr* The IP address of the primary NTP server with which the NetScreen device can synchronize its system clock time.
 - *dom_name* The domain name of the primary NTP server with which the NetScreen device can synchronize its system clock time.
 - **backup1 | backup2**
 - *ip_addr* The IP address of the first (or second) backup NTP server with which the NetScreen device can synchronize its system clock time in case the primary server is not available.
 - *dom_name* The domain name of the first (or second) backup NTP server with which the NetScreen device can synchronize its system clock time in case the primary server is not available.
 - **key-id number** Assigns a key id to the current server for authentication purposes.
 - **preshare key** Assigns a preshared key to the current server for authentication purposes.
 - **key-id number** Assigns a key id to the current server for authentication purposes.
 - **preshare key** Assigns a preshared key to the current server for authentication purposes.

timezone

```
set ntp timezone number1 number2
unset ntp timezone
```

- timezone** Defines the Time Zone, expressed as an integer *number1* between -12 and 12 inclusive. A value of zero denotes GMT (Greenwich Mean Time). *number2* expresses minutes.

Example: The following command sets the Time Zone to Greenwich Mean time:

```
set ntp timezone 0
```

update

```
exec ntp update
```

update	Instructs Updates the time setting on a NetScreen device to synchronize it with the time setting on an NTP server.
---------------	---

Description: Use the **os** commands to display kernel and task information for the operating system of the NetScreen device.

Syntax

get

```
get os { cost | flow | kernel | misc | task name_str }
```

Keywords and Variables

cost

```
get os cost
```

cost Displays the amount of processor time used by elements of the operating system.

flow

```
get os flow
```

flow Displays flow statistics.

kernel

```
get os kernel
```

kernel Displays kernel statistics.

misc

get os misc

misc Displays miscellaneous information.

task

get os task *name_str*

task Displays information on a specified task (*name_str*).

OSPF Commands

Description: Use the **ospf** context to begin configuring OSPF routing protocol for a NetScreen virtual router.

Context Initiation

Initiating the **ospf** context can take up to four steps.

1. Enter the vrouter context by executing the **set vrouter** command.

```
set vrouter vrouter
```

For example:

```
set vrouter trust-vr
```

2. Set the router ID for this virtual routing instance.

```
set router-id { id_num | ip_addr }
```

For example:

```
ns(trust-vr)-> set route-id 172.16.10.10
```

3. Enter the **ospf** context by executing the **set protocol ospf** command.

```
ns(trust-vr)-> set protocol ospf
```

4. Enable OSPF protocol (it is disabled by default).

```
ns(trust-vr/ospf)-> set enable
```

OSPF Commands

The following commands are executable in the **ospf** context.

advertise-def-route	<p>Use the advertise-def-route commands to advertise or display the default route of the current virtual routing instance (0.0.0.0/0) in all areas.</p> <p>Every router has a default route entry, which matches every destination. (Any entry with a more specific prefix overrides the default route entry.)</p> <p>Command options: get, set, unset</p>
area	<p>Use the area commands to configure an area for an OSPF virtual routing instance.</p> <p>An OSPF area is a region that contains a collection of routers or virtual routing instances.</p> <p>Command options: get, set, unset</p>
auto-vlink	<p>Use the auto-vlink commands to direct the local virtual router to automatically create virtual links.</p> <p>Using automatic virtual links replaces the more time-consuming process of creating each virtual link manually. A virtual link is a conveyance that enables two unconnected segments that cannot reach a backbone router to connect with each other.</p> <p>Command options: get, set, unset</p>
config	<p>Use the config command to display all commands executed to configure the OSPF local virtual routing instance.</p> <p>Command options: get</p>
database	<p>Use the database command to display details about the current OSPF link state database.</p> <p>Command options: get</p>
enable	<p>Use the enable commands to enable or disable OSPF from the current routing instance.</p> <p>Command options: set, unset</p>
hello-threshold	<p>Use the hello-threshold commands to set or display the hello threshold. When a neighbor device exceeds this threshold by flooding the virtual router with hello packets, the virtual router drops the extra packets.</p> <p>A Hello packet is a broadcast message that announces the presence of a routing instance on the network.</p> <p>Command options: get, set, unset</p>

interface	Use this command to display all OSPF interfaces on the virtual router. Command options: get
lsa-threshold	Use the lsa-threshold commands to set or display the Link State Advertisement (LSA) threshold. When a neighbor device exceeds this threshold by flooding the virtual router with LSA packets, the virtual router drops the extra packets. Link State Advertisements (LSAs) enable OSPF routers to make device, network, and routing information available for the link state database. Command options: get, set, unset
neighbor	Use the neighbor command to display details about neighbor devices. Command options: get
redistribute	Use the redistribute commands to import routes from a different protocol than the one used by the current virtual routing instance. The types of routing protocols from which to import routes include: <ul style="list-style-type: none">• manually-created routes (static)• routes from BGP (bgp)• routes that have at least one interface with an IP address assigned to it (connected)• routes from RIP (rip)• routes that have already been imported (imported). Command options: get, set, unset
reject-default-route	Use the reject-default-route commands to reject or restore the default route learned from OSPF (0.0.0.0/0) in the current routing instance. Every router has a default route entry in its routing table. This default route matches every destination. (Any entry with a more specific prefix overrides the default route entry.) Command options: get, set, unset
rfc-1583	Use the rfc-1583 commands to use routing table calculation methods consistent with standards specified in the Request For Comments 1583 document. Command options: get, set, unset

<code>routes-redistribute</code>	<p>Use the routes-redistribute command to display details about routes imported from a protocol other than OSPF.</p> <p>Command options: get</p>
<code>rules-redistribute</code>	<p>Use the rules-redistribute command to display conditions set for routes imported from a protocol other than OSPF.</p> <p>Command options: get</p>
<code>statistics</code>	<p>Use the statistics command to display information about Hello packets, link state packets, database descriptions, Shortest Path First (SPF) packets, packets dropped, errors, and other traffic statistics related to the current OSPF virtual routing instance.</p> <p>Command options: get</p>
<code>stub</code>	<p>Use the stub command to display details about a stub area created in the current OSPF virtual routing instance.</p> <p>Command options: get</p>
<code>summary-import</code>	<p>Use the summary-import commands to summarize a route redistribution.</p> <p>After importing a series of routes to the current OSPF routing instance from a router running a different protocol, you can bundle the routes into one generalized (or <i>summarized</i>) address that uses the same network stem of the prefix address. By summarizing multiple addresses, you allow the OSPF routing instance to treat a series of routes as one route, thus simplifying the process.</p> <p>Command options: get, set, unset</p>
<code>vlink</code>	<p>Use the vlink commands to create a virtual link for the current routing instance.</p> <p>A virtual link is a conveyance that allows two segments to connect when the backbone router bridging them cannot reach either segment.</p> <p>Command options: get, set, unset</p>
<code>vneighbor</code>	<p>Use the vneighbor command to display information about a virtual routing instance neighbor.</p> <p>Command options: get</p>

advertise-def-route

Description: Use the **advertise-def-route** commands to advertise or display the default route of the current virtual routing instance (0.0.0.0/0) in all areas.

Every router has a default route entry, which matches every destination. Any route entry with a more specific prefix than the default route entry overrides the default entry.

Before you can execute the **advertise-def-route** commands, you must initiate the **ospf** context. (See ["Context Initiation" on page 319.](#))

Syntax

get

```
get advertise-def-route
```

set

```
advertise-def-route
{
  always metric number [ preserve-metric ] |
  metric number | preserve-metric
}
metric-type { 1 | 2 }
```

unset

```
unset advertise-def-route
```

Keywords and Variables

always

```
set advertise-def-route always { ... }
```

always Directs the routing instance to advertise the default route under all conditions, even if there is no default route in the routing table. If you specify **always**, you must also specify the **metric** parameter; you can optionally specify the **preserve-metric** parameter. If you do not specify **always**, only a non-OSPF active default route is advertised. If you do not specify **always**, you must specify either the **metric** or **preserve-metric** option.

metric

```
set advertise-def-route always metric number metric-type { 1 | 2 }
```

metric Specifies the metric (cost), which indicates the overhead associated with the default route. Enter a number between 1-15. You must specify this parameter if you specify the **always** option.

metric-type

```
set advertise-def-route [ always ] metric number metric-type { 1 | 2 }
```

metric-type Specifies the external route type to determine path preference.

- **1** Directs the routing instance to use a Type 1 route to evaluate the default route. A type 1 route is a comparable route, with a lower cost than a type 2 route.
- **2** Directs the routing instance to use a Type 2 route to evaluate the default route. A type 2 route is a non-comparable route, with a higher cost than a type 1 route.

preserve-metric

```
set advertise-def-route [ always ] preserve-metric metric-type { 1 | 2 }
```

preserve-metric Instructs the NetScreen device to use the original (source) route metric when the route is redistributed.

area

Description: Use the **area** commands to configure an area for an OSPF virtual routing instance.

An OSPF area is a region that contains a collection of routers or virtual routing instances.

Before you can execute the **area** commands, you must initiate the **ospf** context. (See ["Context Initiation" on page 319.](#))

Syntax

get

```
get area { id_num | ip_addr }
```

set

```
set area { id_num | ip_addr }  
  [  
    metric-default-route number |  
    no-summary |  
    nssa |  
    range ip_addr/mask { advertise | no-advertise } |  
    stub |  
    type-default-route { 1 | 2 }  
  ]
```

unset

```
unset area { id_num | ip_addr }  
  [  
    metric-default-route number |  
    no-summary |  
    nssa |
```

```
range ip_addr/mask |  
stub  
type-default-route { 1 | 2 }  
]
```

Keywords and Variables

Variable Parameters

```
get area { id_num | ip_addr }  
set area id_num  
unset area { id_num | ip_addr }
```

ip_addr The IP address that identifies the area.

id_num The OSPF area ID that identifies the area.

metric-default-route

```
set area id_num metric-default-route number  
unset area id_num metric-default-route number
```

metric-default-route (NSSA and stub areas only) Specifies the metric for the advertised default route. The default metric is 1. Enter a number between 1-65535.

no-summary

```
set area id_num no-summary  
unset area id_num no-summary
```

no-summary (NSSA and stub areas only) Prevents summary LSAs from being advertised into the area. By default, summary LSAs are advertised into the area.

nssa

```
set area id_num nssa
unset area id_num nssa
```

nssa Specifies that the area is a "not so stubby area."

range

```
set area id_num range ip_addr/mask { advertise | no-advertise }
unset area id_num range ip_addr/mask
```

range (All areas) Summarizes a specified range of IP addresses in summary LSAs. You can specify multiple ranges for the area. You can specify whether the summarized addresses are advertised inside the area or not with the **advertise** and **no-advertise** keywords.

stub

```
set area id_num stub
```

stub Specifies the area is a stub area.

type-default-route

```
set area id_num type-default-route { 1 | 2 }
unset area id_num type-default-route { 1 | 2 }
```

type-default-route (NSSA area only) Specifies the external metric type for the default route. The default metric type is 1. Specify either 1 or 2.

auto-vlink

Description: Use the **auto-vlink** commands to automatically create or display details about virtual links.

Using automatic virtual links replaces the more time-consuming process of creating each virtual link manually. A virtual link is a conveyance that enables two unconnected segments that cannot reach a backbone router to connect with each other.

Before you can execute the **auto-vlink** commands, you must initiate the **ospf** context. (See ["Context Initiation" on page 319.](#))

Syntax

get

`get auto-vlink`

set

`set auto-vlink`

unset

`unset auto-vlink`

Keywords and Variables

None.

config

Description: Use the **config** command to display all commands executed to configure the OSPF local virtual routing instance.

Before you can execute the **config** command, you must initiate the **ospf** context. (See ["Context Initiation" on page 319.](#))

Syntax

get

get config

Keywords and Variables

None.

database

Description: Use the **database** command to display details about the current OSPF database.

Before you can execute the **database** command, you must initiate the **ospf** context. (See ["Context Initiation" on page 319.](#))

Syntax

get

```
get database
  [ detail ] [ area [ number | ip_addr ] ]
    [ asbr-summary | external | network | nssa-external | router | summary
      [
        adv-router ip_addr |
        self-originate
      ]
    [ link-state-id ip_addr ]
  ]
```

Keywords and Variables

adv-router

```
get database [ ... ] adv-router ip_addr [ ... ]
```

adv-router Displays the LSAs (Link State Advertisements) from the specified advertising router (*ip_addr*).

Example: The following command displays the LSAs from a router with router ID 172.16.10.10:

```
get database adv-router 172.16.10.10
```


area

```
get database [ ... ] area [ number | ip_addr ] [ ... ]
```

area Displays the LSAs in the current area.

Example: The following command displays the LSAs from an area (4):

```
get database area 4
```

detail

```
get database detail [ ... ]
```

detail Displays detailed information.

Example: The following command generates a detailed display of LSAs from an area (4):

```
get database detail area 4
```

external

```
get database [ ... ] external [ ... ]
```

external Displays external LSAs.

Example: The following command displays external LSAs:

```
get database external
```

link-state-id

```
get database { ... } link-state-id ip_addr
```

link-state-id Displays the LSA with a specified link-state ID (*ip_addr*).

Example: The following command generates a detailed display of external LSAs with link-state ID 172.16.1.1:

```
get database detail external link-state-id 172.16.1.1
```

network

```
get database [ ... ] network [ ... ]
```

network Displays the network LSAs.

Example: The following command displays network LSAs:

```
get database network
```

nssa-external

```
get database [ ... ] nssa-external [ ... ]
```

nssa-external Displays the not-so-stubby areas (NSSAs) external LSAs.

Example: The following command displays external LSAs for not-so-stubby areas:

```
get database nssa-external
```

router

```
get database [ ... ] router [ ... ]
```

router Displays router LSAs.

Example: The following command displays router LSAs:

```
get database router
```

self-originate

```
get database [ ... ] self-originate [ ... ]
```

self-originate Displays self-originated LSAs.

Example: The following command displays self-originated LSAs:

```
get database self-originate
```

summary

```
get database [ ... ] summary [ ... ]
```

summary Displays summary LSAs.

Example: The following command displays summary LSAs:

```
get database summary
```

enable

Description: Use the **enable** commands to enable or disable OSPF from the current routing instance.

Before you can execute the **set enable** command, you must initiate the **ospf** context. (See ["Context Initiation" on page 319.](#))

Syntax

set

set enable

unset

unset enable

Keywords and Variables

None.

hello-threshold

Description: Use the **hello-threshold** commands to set or display the hello threshold. When a neighbor device exceeds this threshold by flooding the virtual router with hello packets, the virtual router drops the extra packets. A hello packet is a broadcast message that announces the presence of a routing instance on the network.

Before you can execute the **hello-threshold** commands, you must initiate the **ospf** context. (See ["Context Initiation" on page 319.](#))

Syntax

get

```
get hello-threshold
```

set

```
set hello-threshold number
```

unset

```
unset hello-threshold
```

Keywords and Variables

Variable Parameter

```
set hello-threshold number
```

<i>number</i>	The maximum number of hello packets the virtual router accepts from a neighbor in the hello interval.
---------------	---

Example: The following command sets the maximum number of packets to allow in the hello interval to 1000:

```
ns(trust-vr/ospf)-> set hello-threshold 1000
```

interface

Description: Use this command to display all OSPF interfaces on the virtual router.

Before you can execute the **interface** command, you must initiate the **ospf** context. (See ["Context Initiation" on page 319.](#))

Syntax

get

get interface

Keywords and Variables

None.

lsa-threshold

Description: Use the **lsa-threshold** commands to set or display the Link State Advertisement (LSA) threshold. When a neighbor device exceeds this threshold by flooding the virtual router with LSA packets, the virtual router drops the extra packets.

Link State Advertisements (LSAs) enable OSPF routers to make device, network, and routing information available for the link state database.

Before you can execute the **lsa-threshold** commands, you must initiate the **ospf** context. (See ["Context Initiation" on page 319.](#))

Syntax

get

```
get lsa-threshold
```

set

```
set lsa-threshold number1 number2
```

unset

```
unset lsa-threshold
```

Keywords and Variables

Variable Parameters

set lsa-threshold *number1 number2*

number1 The LSA time interval (in seconds).

number2 The maximum number of LSAs that the virtual router accepts within the time interval expressed by *number1*.

Example: The following command creates an OSPF LSA threshold:

set lsa-threshold 10 30

neighbor

Description: Use the **neighbor** command to display details about neighbor devices.

Before you can execute the **neighbor** command, you must initiate the **ospf** context. (See ["Context Initiation" on page 319.](#))

Syntax

get

get neighbor

Keywords and Variables

None.

redistribute

Description: Use the **redistribute** commands to import known routes from a router running a different protocol than the current virtual routing instance.

The types of routers from which to import routes include:

- routers with manually created routes (**static**)
- routers running BGP (**bgp**)
- routers that have at least one interface with an IP address assigned to it (**connected**)
- routers with routes that have already been imported (**imported**)
- routers running RIP (**rip**)

Before you can execute the **redistribute** commands, you must initiate the **ospf** context. (See [“Context Initiation” on page 319.](#))

Syntax

get

```
get routes-redistribute
get rules-redistribute
```

set

```
set redistribute route-map string protocol
    { bgp | connected | imported | rip | static }
```

unset

```
unset redistribute route-map name_str protocol
    { bgp | connected | imported | rip | static }
```

Keywords and Variables

protocol

```
set redistribute route-map string protocol { ... }
```

- protocol** Specifies routing protocol. The route map can use the protocol type to determine whether to forward or deny an incoming packet.
- **bgp** specifies that the route map performs an action only on BGP routes in the subnetwork.
 - **connected** specifies that the route map performs an action only on routes sent from a router that has at least one interface with an IP address assigned to it.
 - **imported** specifies that the route map performs an action only on imported routes in the subnetwork.
 - **rip** specifies that the route map performs an action only on RIP routes in the subnetwork.
 - **static** specifies that the route map performs an action only on static routes in the subnetwork.

Example: The following command redistributes a route that originated on a router that has at least one interface with an IP address assigned to it:

```
ns(trust-vr/ospf)-> set redistribute route-map map1 protocol connected
```

route-map

```
set redistribute route-map string protocol { ... }
```

route-map Identifies the route map that indicates the path for which the route should be imported.

Example: The following command redistributes a route that originated from a BGP routing domain into the current OSPF routing domain:

```
ns(trust-vr/ospf)-> set redistribute route-map map1 protocol bgp
```

reject-default-route

Description: Use the **reject-default-route** commands to reject or restore the default route learned from OSPF (0.0.0.0/0).

Every router has a default route entry in its routing table. This default route matches every destination. (Any entry with a more specific prefix overrides the default route entry.)

Before you can execute the **reject-default-route** commands, you must initiate the **ospf** context. (See ["Context Initiation" on page 319.](#))

Syntax

get

`get reject-default-route`

set

`set reject-default-route`

unset

`unset reject-default-route`

Keywords and Variables

None.

rfc-1583

Description: Use the **rfc-1583** commands to use routing table calculation methods consistent with standards specified in the Request For Comments 1583 document.

Before you can execute the **rfc-1583** commands, you must initiate the **ospf** context. (See ["Context Initiation" on page 319.](#))

Syntax

get

`get rfc-1583`

set

`set rfc-1583`

unset

`unset rfc-1583`

Keywords and Variables

None.

routes-redistribute

Description: Use the **routes-redistribute** command to display details about routes imported from a protocol other than OSPF.

Before you can execute the **routes-redistribute** command, you must initiate the **ospf** context. (See ["Context Initiation" on page 319.](#))

Syntax

get

`get routes-redistribute`

Keywords and Variables

None.

rules-redistribute

Description: Use the **rules-redistribute** command to display conditions set for routes imported from a protocol other than OSPF.

Before you can execute the **rules-redistribute** command, you must initiate the **ospf** context. (See ["Context Initiation" on page 319.](#))

Syntax

get

`get rules-redistribute`

Keywords and Variables

None.

statistics

Description: Use the **statistics** command to display information about the following objects associated with an OSPF virtual routing instance:

- Hello Packets
- Link State Requests
- Link State Acknowledgments
- Link State Updates
- Database Descriptions
- Areas Created
- Shorted Path First Runs
- Packets Dropped
- Errors Received
- Bad Link State Requests

Before you can execute the **statistics** command, you must initiate the **ospf** context. (See [“Context Initiation” on page 319.](#))

Syntax

get

get statistics

Keywords and Variables

None.

stub

Description: Use the **stub** command to display details about a stub area created for the current OSPF virtual routing instance.

Before you can execute the **stub** command, you must initiate the **ospf** context. (See ["Context Initiation" on page 319.](#))

Syntax

get

```
get stub [ ip_addr ]
```

Keywords and Variables

Variable Parameters

```
get stub ip_addr
```

ip_addr Identifies the stub area.

Example: The following command displays details about a stub area created on the current OSPF virtual routing instance:

```
ns(trust-vr/ospf)-> get stub 192.168.20.20
```

summary-import

Description: Use the **summary-import** commands to summarize a route redistribution.

After importing a series of routes to the current OSPF routing instance from a router running a different protocol, you can bundle the routes into one generalized (or *summarized*) address that uses the same network stem of the prefix address. By summarizing multiple addresses, you allow the OSPF routing instance to treat a series of routes as one route, thus simplifying the process.

Before you can execute the **summary-import** commands, you must initiate the **ospf** context. (See ["Context Initiation" on page 319.](#))

Syntax

get

```
get summary-import
```

set

```
set summary-import ip ip_addr/mask [ tag { ip_addr | id_num } ]
```

unset

```
unset summary-import ip ip_addr/mask
```

Keywords and Variables

ip

```
set summary-import ip ip_addr/mask [ ... ]  
unset summary-import ip ip_addr/mask
```

ip The summarized prefix, consisting of an address (*ip_addr*) and network mask (*mask*) encompassing all the imported routes.

tag

```
set summary-import ip ip_addr/mask tag { ip_addr | id_num }
```

tag A value that acts as an identifier for the summarized prefix. The virtual router uses this identifier when advertising a new external LSA.

Example: The following command summarizes a set of imported routes under one route (20):

```
ns(trust-vr/ospf)-> set summary-import ip 2.1.1.0/24 tag 20
```

Description: Use the **vlink** commands to create a virtual link for the current routing instance.

A virtual link is a conveyance that allows two segments to connect when the backbone router bridging them cannot reach either segment.

Before you can execute the **vlink** command, you must initiate the **ospf** context. (See ["Context Initiation" on page 319.](#))

Syntax

get

```
get vlink
```

set

```
set vlink area-id { id_num1 | ip_addr } router-id { id_num2 | ip_addr }  
  [  
    authentication  
    {  
      active-md5-key-id id_num |  
      md5 key_str [ key-id id_num ] |  
      password pswd_str  
    } |  
    dead-interval number |  
    hello-interval number |  
    retransmit-interval number |  
    transit-delay number  
  ]
```

unset

```
unset vlink area-id { id_num1 | ip_addr } router-id { id_num2 | ip_addr }
[
  authentication
  [
    active-md5-key-id |
    md5 [ key-id id_num ] ]
  ] |
  dead-interval number |
  hello-interval number |
  retransmit-interval number |
  transit-delay number
]
```

Keywords and Variables

area-id

```
set vlink area-id id_num1 { ... }
unset vlink area-id id_num1 { ... }
```

area-id Specifies the ID or IP address of the area through which the virtual link is connected.

authentication

```
set vlink { ... } authentication
{ active-md5-key-id | md5 key_str [ key-id id_num ] | password pswd_str }
unset vlink { ... } authentication [ active-md5-key-id | md5 [ key-id id_num ] ]
```

authentication Specifies the authentication method, including MD5 key string, the key identifier number (the default is 0), and password. You can specify more than one MD5 key with different key identifier numbers (between 0-255). If there are multiple MD5 keys configured, you can use the **active-md5-key-id** option to select the key identifier of the key to be used for authentication.

dead-interval

```
set vlink { ... } dead-interval number  
unset vlink { ... } dead-interval number
```

dead-interval *Specifies the maximum amount of time that the NetScreen device waits, after it stops receiving packets from the neighbor, before classifying the neighbor as offline.*

hello-interval

```
set vlink { ... } hello-interval number  
unset vlink { ... } hello-interval number
```

hello-interval *Specifies the amount of time in seconds that elapse between instances of the interface sending Hello packets to the network announcing the presence of the interface.*

retransmit-interval

```
set vlink { ... } retransmit-interval number  
unset vlink { ... } retransmit-interval number
```

retransmit-interval *Specifies the amount of time (in seconds) that elapses before the interface resends a packet to a neighbor that did not acknowledge a previous transmission attempt for the same packet.*

router-id

```
set vlink area-id id_num1 router-id id_num2  
unset vlink area-id id_num1 router-id id_num2
```

router-id *Specifies the ID or IP address of the router at the other end of the virtual link.*

Example: The following command creates a virtual link using an area of 0.0.0.1 for router with an ID of 10.10.10.20:

```
ns(trust-vr/ospf)-> set vlink area-id 0.0.0.1 router-id 10.10.10.20
```

transit-delay

```
set vlink { ... } transit-delay number  
unset vlink { ... } transit-delay number
```

transit-delay *Specifies the amount of time (in seconds) that elapses before the NetScreen device advertises a packet received on the interface.*

vneighbor

Description: Use the **vneighbor** command to display information about a neighbor on the virtual link.

Before you can execute the **vneighbor** command, you must initiate the **ospf** context. (See ["Context Initiation" on page 319.](#))

Syntax

get

get vneighbor

Keywords and Parameters

None.

performance

Description: Use the performance command to retrieve performance information for the NetScreen device.

You can display information for the following:

- CPU utilization
- The session ramp-up rate

Syntax

get

```
get performance { cpu | session } [ detail ]
```

Keywords and Variables

cpu

```
get performance cpu [ detail ]
```

cpu	Displays the current CPU utilization rate for the last minute, the last 5 minutes, and the last fifteen minutes.
------------	--

detail

```
get performance { cpu | session } detail
```

detail	Displays the CPU utilization or session ramp-up rate for last 60 seconds, the last 60 minutes, and the last 24 hours.
---------------	---

session

get performance session [detail]

session Displays the number of sessions added (ramp-up rate) for the last minute, the last 5 minutes, and the last fifteen minutes. It does not display the total number of sessions or the number of deleted sessions.

ping

Description: Use the **ping** command to check the network connection to another system.

Note: An extended **ping** (using the **from** option) pings a host on the Untrusted network from any existing MIP, or from the Trusted interface IP address. The syntax for specifying a MIP is **mip** ip_addr (see example in **from** keyword description).

Syntax

```
ping [ ip_addr | name_str ]  
      [ count number [ size number [ time-out number ] ] ]  
      [ from interface ]
```

Keywords and Variables

Variable Parameters

```
ping [ ip_addr | name_str ] [ ... ]
```

ip_addr | name_str Pings the host at address (*ip_addr*) or with name (*name_str*).

Example: The following command pings a host with IP address 172.16.11.2:

```
ping 172.16.11.2
```

count

```
ping [ ip_addr | name_str ] count number [ ... ]
```

count The ping count (*number*).

Example: The following command pings a device at 10.100.2.171 with a ping count of 3:

```
ping 10.100.2.171 count 3
```

from

ping [*ip_addr* | *name_str*] [...] **from** *interface*

from The source interface (*interface*) for an extended ping. For more information on interfaces, see ["Interface Names" on page A-I](#).
Defines the source IP to which the ping will reply. Because this destination is on the untrusted side, the source IP can only be the Mapped IP address or an untrusted interface IP address.

Examples: The following command pings a device at 10.100.2.11 with a ping count of 4 from the ethernet1 interface:

```
ping 10.100.2.11 count 4 from ethernet1
```

The following command pings a host with IP address 192.168.11.2 and sends the results to IP address 10.1.1.3:

```
ping 192.168.11.2 from mip 10.1.1.3
```

size

ping [*ip_addr* | *name_str*] **count** *number* **size** *number* [...]

size The packet size (*number*) for each ping.

time-out

ping [*ip_addr* | *name_str*] **count** *number* **size** *number* **time-out** *number*

time-out The ping timeout in seconds (*number*).

Example: The following command pings a device at 10.100.2.11.

- Ping count of 4
- Packet size 1000
- Ping timeout of three seconds:

```
ping 10.100.2.11 count 4 size 1000 time-out 3
```

Definition: Use the **pki** commands to manage public-key infrastructure (PKI).

The term Public Key Infrastructure (PKI) refers to the hierarchical structure of trust required for public key cryptography. Using PKI, the NetScreen device verifies the trustworthiness of a certificate by tracking a path of certificate authorities (CAs) from the one issuing your local certificate back to a root authority of a CA domain.

The **pki** commands perform the following tasks:

- Manage PKI objects.
- Create new RSA key pairs and acquire a certificate.
- Verify the certificate received from the communication peer.
- Acquire CRLs.
- Configure PKI-related operations, such as verification of certificate revocation.

Syntax

exec

```
exec pki
{
  convert-cert |
  dsa new-key number [ & ] |
  rsa new-key number [ & ] |
  x509
  {
    install-factory-certs name_str |
    pkcs10 |
    scep
    {
      cert id_num |
      key { id_num | last-key } |
      renew id_num |
    }
  }
}
```

```
    } |  
    tftp ip_addr { cert-name name_str | crl-name name_str }  
  }  
}
```

get

```
get pki  
{  
  authority { id_num | default }  
  {  
    cert-path |  
    cert-status |  
    scep  
  } |  
  ldap |  
  pre-prime |  
  src-interface |  
  x509  
  {  
    cert id_num |  
    cert-fqdn |  
    cert-path |  
    crl-refresh |  
    dn |  
    list { ca-cert | cert | crl | key-pair | local-cert | pending-cert } |  
    pkcs10 |  
    raw-cn |  
    send-to  
  }  
}
```

set (authority)

```
set pki authority { id_num | default }
{
  cert-path { full | partial } |
  cert-status
  {
    crl
    {
      refresh { daily | default | monthly | weekly } |
      server-name { ip_addr | dom_name } |
      url url_str
    }
    ocsp
    {
      cert-verify id id_num |
      not-verify-revoke |
      url url_str
    } |
    revocation-check { crl [ best-effort ] | ocsp [ best-effort ] | none }
  }
  scep
  {
    authentication { failed | passed } |
    ca-cgi string |
    ca-id name_str |
    challenge pswd_str |
    current |
    mode { auto | manual } |
    polling-int number |
    ra-cgi string |
    renew-start number
  }
}
```

set (ldap)

```
set pki ldap
{
  crl-url url_str |
  server-name { name_str | ip_addr }
}
```

set (pre-prime)

```
set pki pre-prime number
```

set (src-interface)

```
set pki src-interface interface
```

set (x509)

```
set pki x509
{
  cert-fqdn string |
  default
  {
    crl-refresh { daily | default | monthly | weekly } |
    send-to string
  } |
  dn
  {
    country-name name_str |
    email string |
    ip ip_addr |
    local-name name_str |
    name name_str |
    org-name name_str |
    org-unit-name name_str |
  }
}
```



```
    phone string |
    state-name name_str
  } |
  friendly-name string |
  raw-cn enable |
  renew id_num
}
```

unset

```
unset pki
{
  authority { id_num | default }
  {
    cert-path |
    cert-status
    {
      crl { refresh | server-name | url } |
      cert-verify
      revocation-check
    } |
    scep
    {
      ca-cgi |
      ca-id |
      challenge |
      current |
      mode |
      polling-int |
      ra-cgi |
      renew-start
    }
  } |
  ldap
  {
```

```
    crl-url |
    server-name
  } |
pre-prime |
src-interface |
x509
{
  cert-fqdn |
  default { cert-path | crl-refresh | send-to } |
  dn
  {
    country-name |
    email |
    ip |
    local-name |
    name |
    org-name |
    org-unit-name |
    phone |
    state-name
  }
  friendly-name id_num |
  raw-cn
}
```

Keywords and Variables

authentication

```
set pki authority { ... } scep authentication { failed | passed } [ id_num ]
```

authentication Sets the result of the CA certificate authentication, **failed** or **passed**. The *id_num* value identifies a pending certificate created during a SCEP operation.

Example: The following command sets the result of a CA certificate authentication to **passed**:

```
set pki authority default scep authentication passed
```

authority

```
get pki authority { id_num | default } { ... }  
set pki authority { id_num | default } { ... }  
unset pki authority { id_num | default } { ... }
```

authority Defines how the NetScreen device uses the CA's authorization services. The *id_num* parameter is the identification number of the CA certificate.
The **default** switch directs the device to use the authority configuration (used when the CA certificate does not reside locally).

Example: The following command instructs the NetScreen device to check for certificate revocation on a daily basis:

```
set pki authority default cert-status crl refresh daily
```

cert-path

```
get pki authority { id_num | default } cert-path
set pki authority { id_num | default } cert-path { full | partial }
unset pki authority id_num cert-path
```

cert-path

Defines the X509 certificate path validation level.

When the device verifies a certificate, it builds a certificate chain from certificates received from the peer and the certificate stored locally. Certificates loaded locally are considered "trusted".

- **full** Directs the NetScreen device to validate the certificate chain to the root. (The last certificate in the certificate chain must be a self-signed CA certificate.)
- **partial** Specifies partial path validation. (The last certificate in the certificate chain may be any locally-stored certificate.)

In either case, the last certificate in the chain must come from local storage. You can set this certificate path validation level for a CA.

Example: The following command defines the certificate path validation level as full:

```
set pki authority default cert-path full
```

cert-status

```
get pki authority { id_num | default } cert-status
set pki authority { id_num | default } cert-status { ... }
unset pki authority { id_num | default } cert-status { ... }
```

cert-status

Defines how the NetScreen device verifies the revocation status of a certificate.

- **crl** Configures Certificate Revocation List (CRL) parameters.
 - **refresh** Determines how often (**daily**, **monthly**, or **weekly**) the NetScreen device updates the CRL before the CRL expires. The **default** option uses the validation date decided by the CRL.
 - **server-name** { *ip_addr:port_num* | *dom_name* } Specifies the server by IP address and port number, or by domain name.
 - **url** *url_str* Specifies the URL for accessing the CRL.

- **ocsp** Configures Online Certificate Status Protocol (OCSP) parameters.
 - **cert-verify id** *number* Identifies the certificate to use when verifying the OCSP response.
 - **not-verify-revoke** Disables verification of revocation status on the OCSP signing certificate.
 - **url** *url_str* Specifies the URL for accessing the OCSP responder.
- **revocation-check** Specifies how the NetScreen device checks certificates to see if they are currently revoked.
 - **crl** Specifies that the device uses CRL to check certificate status.
 - **none** Specifies that the device does not perform a check of certificate status.
 - **ocsp** Specifies that the device uses OCSP to check certificate status.
 - **best-effort** Specifies that the device can use a certificate for which there is no revocation information. This option is useful when CRL retrieval is not practical. For example, in some environments the CRL server is only accessible through a tunnel; however, the CRL information is necessary to build the tunnel originally. When you use the **best-effort** setting, it is advisable to check the event log periodically. The device should accept a certificate without revocation information only when no revocation information is available. Repeatedly failing to get revocation information for a certificate usually indicates improper configuration.

Example: The following command directs the NetScreen device to use the CRL to check certificate status:

```
set pki authority default cert-status revocation-check crl
```

cert-verify id

```
set pki authority id_num1 cert-status ocsp cert-verify id id_num2  
unset pki authority id_num cert-status ocsp cert-verify
```

- | | |
|-----------------------|---|
| cert-verify id | Identifies a locally-stored certificate the NetScreen device uses to verify the signature on an OCSP responder. <ul style="list-style-type: none">• <i>id_num1</i> Identifies the CA certificate that issued the certificate being verified.• <i>id_num2</i> Identifies the locally stored certificate the device uses to verify the signature on the OCSP response. |
|-----------------------|---|

convert-cert

```
exec pki convert-cert
```

convert-cert Converts vsys certificate (for versions prior to ScreenOS 3.0.0) to use the internal vsys identifier in ScreenOS 3.0.0 and above.

dsa new-key

```
exec pki dsa new-key number
```

dsa new-key Generates a new DSA public/private key pair with a specified bit length (*number*). Key length is 512, 786, 1024, or 2048.

ldap

```
get pki ldap  
set pki ldap { ... }  
unset pki ldap { ... }
```

ldap Specifies settings for the LDAP server, when the CA certificate associated with the server is not in the device.

- **crl-url** *url_str* Sets the default LDAP URL for retrieving the certificate revocation list (CRL).
- **server-name** { *name_str* | *ip_addr:port_num* } Defines the fully-qualified domain name or IP address and port number of the server.

Example: The following command assigns 162.128.20.12 as the server's IP address:

```
set pki ldap server-name 162.128.20.12
```

pre-prime

```
get pki pre-prime
set pki pre-prime number
unset pki pre-prime
```

pre-prime

The **get** command displays:

- The number of pre-calculated primes for every key type and key length combination. The key type can be DSA or RSA and the key length can be 1024 or 2048 bits depending on the NetScreen platform*.
- The number of currently available pairs of prime numbers for every key type and key length combination.
- Ongoing prime calculation for a key type and key length combination and the number of attempts already made.

The **set** command instructs the NetScreen device to generate a specific number of pre-calculated primes to store in memory.

The **unset** command reverts the NetScreen device to the default number of pre-calculated primes. The default number of pre-calculated primes is platform specific. For more information, refer to the Specification Sheet for your NetScreen product.

* NetScreen appliances generate 1024-bit primes. NetScreen systems generate 1024- and 2048-bit primes. For more information, refer to the Specification Sheet for your NetScreen product.

rsa new-key

```
exec pki rsa new-key number [ & ]
```

rsa new-key

Generates a new RSA public/private key pair with a specified bit length (*number*). Key length is 512, 786, 1024, or 2048.

The **&** switch directs the device to perform key generation in the background, without waiting for the result. Without this switch, the device waits up to 100 seconds.

scep

```
exec pki x509 scep { cert id_num | key { id_num | last-key } | renew }
get pki authority { id_num | default } scep
set pki authority { id_num | default } scep { ... }
unset pki authority { id_num | default } scep { ... }
```

scep

Defines Simple Certificate Enrollment Protocol (SCEP) parameters.

- **authentication** { **passed** | **failed** } [*id_num*] sets the result of the CA authentication, failed or passed. The *id_num* value identifies a defined key pair.
- **ca-cgi** *url_str* Specifies the path to the CA's SCEP server.
- **ca-id** *string* Specifies the identity of the CA's SCEP server.
- **cert-id** *id_num* Directs the NetScreen device to retrieve the final certificate for a pending certification.
- **challenge** *pswd_str* Specifies the Challenge password.
- **current** Directs the NetScreen device to use the SCEP associated with a CA as default.
- **key** *id_num* Directs the device to acquire a certificate for the specified key pair. The *id_num* parameter specifies the ID of a specific key pair. The **last_key** parameter specifies the most recently-created key pair.
- **mode** { **auto** | **manual** } Specifies the authentication mode to authenticate the certificate.
- **polling-int** *number* Determines the retrieval polling interval (in minutes). The default value is 0 (none).
- **ra-cgi** *url_str* Specifies the CGI path to the RA's SCEP server.
- **renew** *id_num* Directs the device to renew the specified certificate (*id_num*).
- **renew-start** Set the number of days before the certificate expiration date when you want the NetScreen device to request the renewal of the certificate.

Example: The following command sets the SCEP Challenge password to "swordfish:"

```
set pki authority default scep challenge swordfish
```


Example: The following command uses the SCEP setting for CA 123 as the default:

```
set pki authority 123 scep current
```

send-to

```
get pki x509 send-to
set pki x509 default send-to string
unset pki x509 default send-to
```

send-to Specifies or displays the email destination (*string*) to send the x509 certificate request file.

src-interface

```
get pki src-interface
set pki src-interface
unset pki src-interface
```

src-interface Displays, configures or removes the source interface the NetScreen device uses to send PKI traffic.

x509

```
exec pki x509 { ... }
get pki x509 { ... }
set pki x509 { ... }
unset pki x509 { ... }
```

x509 Specifies settings for x509 certificates, displays certificate information, and performs various operations related to x509 PKI object.

- **cert *id_num*** Displays information on the specified certificate.
- **cert-fqdn *string*** Configures the Fully-Qualified Domain Name (FQDN). PKI uses this value in the certificate subject alt name extension.

- **default** Specifies settings for the CA whose certificate is not locally configured.
 - **crl-refresh** Sets or displays the refreshment frequency (**daily**, **monthly**, or **weekly**) of the X.509 CRL. The **default** option uses the expiration date in each CRL.
 - **send-to** *string* Assigns the e-mail address to which the NetScreen device sends the PKCS10 certificate request file.
- **dn** Specifies or displays the name that uniquely identifies a requesting certificate.
 - **country-name** *name_str* Sets the country name.
 - **email** *string* Sets the e-mail address.
 - **ip** *ip_addr* Sets the IP address.
 - **local-name** *string* Sets the locality.
 - **name** *string* Sets the name in a common name field.
 - **org-name** *string* Sets the organization name.
 - **org-unit-name** *string* Sets the organization unit name.
 - **phone** *string* Sets a contact phone number as the X.509 certificate subject name of the NetScreen device.
 - **state-name** *string* Sets the state name as the X.509 certificate subject name.
- **friendly-name** *name_str id_num* A friendly name (*name_str*) for the certificate (*id_num*).
- **install-factory-certs** *name_str* Loads a specified factory pre-defined certificate.
- **list** Displays the X.509 object list.
 - **ca-cert** Displays all CA certificates.
 - **cert** Displays all X.509 certificates.
 - **key-pair** Displays all key pairs for which there is no certificate.
 - **crl** Displays all Certificate Revocation Lists (CRLs).
 - **local-cert** Displays all local certificates.
 - **pending-cert** Displays all pending certificates.
- **pkcs10** Displays a PKCS10 file (an X.509 certificate request) for a key pair.

- **raw-cn enable** Enables the raw common name (CN) or displays its current status.
- **scep**
 - **cert** *id_num* Initiates Simple Certificate Enrollment Protocol (SCEP) operation to retrieve certificates from a certificate authority server. The *id_num* parameter is the identification number of the pending certificate.
 - **key** { *id_num* | **last-key** } Initiates SCEP operation to obtain a certificate for a key pair. The variable *id_num* identifies the key pair and **last-key** specifies to obtain a certificate for the most recently created key pair.
 - **renew** *id_num* Initiates SCEP operation to renew an existing certificate. The variable *id_num* identifies the existing certificate to renew.
- **tftp** *ip_addr* Uploads the specified certificate (**cert-name** *name_str*) or CRL file (**crl-name** *name_str*) for the specified TFTP server at IP address *ip_addr*.

Examples: The following command specifies the destination e-mail address where the NetScreen device sends the PKCS10 certificate request:

```
set pki x509 default send-to caServer@somewhere.com
```

The following command refreshes the certificate revocation list on a daily basis:

```
set pki x509 default crl-refresh daily
```

The following command defines a distinguished name for **Ed Jones**, who works in marketing at Juniper Networks in Santa Clara, California:

```
set pki x509 dn country-name "US"  
set pki x509 dn state-name CA  
set pki x509 dn local-name "santa clara"  
set pki x509 dn org-name "juniper networks"  
set pki x509 dn org-unit-name marketing  
set pki x509 dn name "ed jones"
```

Defaults

The RSA key length is set to 1024 bits.

Requesting a CA Certificate

You use the **set pki**, **get pki**, and **exec pki** commands to request an x509 CA certificate from a certificate authority. The following commands provide a typical example:

1. Specify a certificate authority CA CGI path.

```
set pki auth default scep ca-cgi "http://pilotsiteipsec.verisign.com/cgi-bin/  
pkiclient.exe"
```

Note: The Common Gateway Interface (CGI) is a standard way for a web server to pass a user request to an application program, and to receive data back. CGI is part of the web's Hypertext Transfer Protocol (HTTP).

2. Specify a registration authority RA CGI path

```
set pki auth default scep ra-cgi "http://pilotsiteipsec.verisign.com/cgi-bin/  
pkiclient.exe"
```

Note: You must specify an RA CGI path even if the RA does not exist. If the RA does not exist, use the value specified for the CA CGI.

3. Generate an RSA key pair, specifying a key length of 1024 bits.

```
exec pki rsa new 1024
```

4. Initiate the SCEP operation to request a local certificate.

```
exec pki x509 scep key last-key
```

5. If this is the first attempt to apply for a certificate from this certificate authority, a prompt appears presenting a fingerprint value for the CA certificate. (Otherwise, go on to Step 6.)

After verification of the fingerprint, allow the operation to continue by executing the following command:

```
set pki auth default scep auth passed
```

You must specify an RA CGI path even if the RA does not exist. If the RA does not exist, use the value specified for the CA CGI.

6. If the device does not approve the certificate automatically, contact your certificate authority administrator to approve the local certificate request.
7. (Optional) Display a list of pending certificates. This allows you to see and record the ID number identifying the pending certificate.

```
get pki x509 list pending-cert
```

8. (Optional) Obtain the local certificate from the CA (using the ID number obtained in Step 7) to identify the certificate. In this example, the certificate number is 1001.

```
exec pki x509 scep cert 1001
```

policy

Description: Use the **policy** commands to define policies to control network and VPN traffic.

A *policy* is a set of rules that determines how traffic passes between security zones (interzone policy), between interfaces bound to the same zone (intrazone policy), and between addresses in the Global zone (global policy). When a NetScreen device attempts to pass a packet from one zone to another, between two interfaces bound to the same zone, or between two addresses in the Global zone, the NetScreen device checks its policy lists for a policy to permit such traffic. For example, to allow traffic to pass from one security zone to another, you must configure a policy that permits zone A to send traffic to zone B. To allow traffic originating in zone B to flow to zone A, you must configure another policy permitting traffic from zone B to zone A.

Executing the **set policy id pol_num** command without specifying further options places the CLI within the context of an existing policy. For example, the following commands define a policy with ID number 1 and then enter the “policy:1” context to add a second service:

```
ns-> set policy id 1 from trust to untrust host1 host2 HTTP permit
ns-> set policy id 1
ns(policy:1)-> set service FTP
```

After you enter a policy context, all subsequent command executions modify the specified policy (policy:1 in this example). To save your changes, you must first exit the policy context, and then enter the **save** command:

```
ns(policy:1)-> exit
ns-> save
```

You can also use the **set policy id pol_num** command with additional options to modify an existing policy. For example, the following commands add a Deep Inspection extension to policy 1:

```
ns-> set policy id 1 from trust to untrust host1 host2 HTTP permit
ns-> set policy id 1 attack HIGH:HTTP:SIGS action close
```

Note: The above example adds a Deep Inspection extension that was not present in the original policy. After you enter a policy context, you cannot add a Deep Inspection extension if one does not already exist in the original policy.

Syntax

exec

```
exec policy verify [ from zone [ to zone ] | global | to zone ]
```

get

```
get policy  
[  
  all |  
  from zone1 to zone2 |  
  [ global ] id pol_num  
]
```

get (within a policy context)

```
get configuration
```

set

```
set policy  
[ global ]  
[ id pol_num1 ]  
[ top | before pol_num2 ]  
[ name name_str ]  
[ from zone1 to zone2 ]  
src_addr dst_addr svc_name  
[  
  nat  
    [ src [ dip-id id_num ] ]  
    [ dst ip addr1 [ addr2 | port port_num ] ]  
  ]  
  {
```

```
permit |
tunnel { l2tp tunn_str | vpn-group id_num } |
tunnel vpn tunn_str [ l2tp tunn_str | pair-policy pol_num ]
}
    [ auth [ server name_str ] | webauth ]
    [
        group-expression string |
        user name_str | user-group name_str
    ]
] |
deny
    [ schedule name_str ]
    [ log [ alert ] ]
        [ count [ alarm id_num1 id_num2 ] ]
        [ no-session-backup ]
    [ url-filter ]
        [ traffic { gbw number }
            { priority number }
            { mbw [ number ]
                dscp { disable | enable }
            }
        ]
    [ attack string action string | av name_str ]
}
set policy move pol_num1 { before pol_num2 | after pol_num3 }
set policy default-permit-all
```

set policy id number

```
set policy [ global ] id pol_num disable
set policy [ global ] id pol_num application svc_name
set policy [ global ] id pol_num attack string action string
set policy [ global ] id pol_num av name_str
```


set (within a policy context)

```
set
{
  attack string |
  av name_str |
  count [ alarm number1 number2 ] |
  di-alert-disable |
  di-severity { info | low | medium | high | critical } |
  dst-address | src-address
    { name_str | negate } |
  log [ alert ] |
  name name_str |
  service svc_name |
  src-address { name_str | negate }
}
```

unset

```
unset policy { pol_num | id pol_num }
[
  application |
  attack |
  av name_str |
  disable
]
unset policy default-permit-all
```

unset (within a policy context)

```
unset attack string
unset av name_str
unset count
unset { dst-address | src-address } { name_str | negate }
unset { ims-alert | ims-log }
```

```
unset log
unset name
unset service svc_name
unset severity
```

Keywords and Variables

all

```
get policy all
```

all Displays information about all security policies.

application

```
set policy [ global ] id pol_num application svc_name
```

application Defines the type of Layer 7 application associated with a Layer 3 service and Layer 4 port number. This is particularly important for defining the Layer 7 application for custom services so that the NetScreen device can properly inspect such traffic for attack signatures and anomalies.

The **ignore** option, which appears near the end of the list of application choices, instructs the NetScreen device to ignore the application type typically associated with a predefined service and port number. Using the **ignore** option instructs the NetScreen device not to scan the packet payload and can prevent the NetScreen device from attempting to parse one type of traffic when it is actually another type—such as the case with LDAP and H.323 traffic, both of which use TCP port 389.

The **none** option, which also appears near the end of the list of application choices, instructs the NetScreen device to use the default setting. Choosing **none** is the equivalent to entering the CLI command: **unset policy id *id_num* application**.

Example: The following command identifies the Layer 7 application for policy ID 1 as FTP:

```
set policy id 1 application FTP
```

attack

```
set policy { ... } attack string action string
set attack string
unset policy { pol_num | id pol_num } attack
unset attack string
```

- attack string** Inspects traffic to which the policy applies for attack objects in the specified attack object group. Attack objects can be stateful signatures or protocol anomalies. If the NetScreen device detects an attack object, it then performs the specified **action**, which can be one of the following:
- **Close** The NetScreen device logs the event, severs the connection, and sends TCP RST packets to both the client and server.
 - **Close Client** The NetScreen device logs the event, severs the connection, and sends a TCP RST packet to the client.
 - **Close Server** The NetScreen device logs the event, severs the connection, and sends a TCP RST to the server.
 - **Drop** The NetScreen device logs the event and severs the connection without sending either the client or the server TCP RST packets.
 - **Drop Packet** The NetScreen device logs the event and drops the packet containing the attack object, but it does not sever the connection.
 - **Ignore** The NetScreen device logs the event and stops checking—or ignores—the remainder of the connection.
 - **None** The NetScreen device logs the event but takes no action.

Example: The following commands define a policy to check for attack objects in the CRITICAL:HTTP:ANOM, CRITICAL:HTTP:SIGS, HIGH:HTTP:ANOM, and HIGH:HTTP:SIGS attack object groups in HTTP traffic from any host in the Untrust zone to webserver1 in the DMZ zone. If the NetScreen device detects any attack objects, it then severs the connection and sends webserver1 a TCP RST so it can clear its resources:

```
ns-> set policy id 1 from untrust to dmz any webserver1 http permit attack
      CRITICAL:HTTP:ANOM action close server
ns-> set policy id 1
ns(policy:1)-> set attack CRITICAL:HTTP:SIGS
```

```
ns(policy:1)-> set attack HIGH:HTTP:ANOM
ns(policy:1)-> set attack HIGH:HTTP:SIGS
```

av

```
set policy { ... } av name_str
set av name_str
unset policy { pol_num | id pol_num } av name_str
unset av name_str
```

av *name_str* Sends HTTP or SMTP traffic to which the policy applies to the specified antivirus (AV) scanner, which examines the data for viruses. If it finds a virus, the AV scanner quarantines the infected data for further study and returns the SMTP or HTTP file—without the infected data—to the NetScreen device, which then forwards the file to the intended recipient

Example: The following command instructs the NetScreen device to forward SMTP traffic originating from the remote mail server "r-mail1" in the Untrust zone and destined for the local mail server "mail1" in the DMZ zone to an AV scanner named "av1":

```
set policy id 1 from untrust to dmz r-mail1 mail1 smtp permit av av1
```

auth

```
set policy { ... } auth [ ... ]
```

auth Requires the user to provide a login name and password to authenticate his or her identity before accessing the device and crossing the firewall.

- **server** *name_str* Identifies the authentication server (*name_str*).
- **group-expression** *string* Identifies users according to an expression (*string*).
- **user** *name_str* Identifies a user (*name_str*).
- **user-group** *name_str* Identifies a user group (*name_str*).

Example: The following command invokes user authentication.

- Permits any kind of traffic from any address in the Trust zone to any address in the Untrust zone
- Uses an authentication server named wc-server

```
set policy from trust to untrust any any any permit auth server wc-server
```

before

```
set policy before pol_num1 { ... }
```

before Specifies the position of the policy before another policy (*pol_num*) in the access control list (ACL).

Example: The following command creates a new policy with ID number 3 and positions it before the policy with ID number 2:

```
set policy id 3 before 2 from trust to untrust any any any permit
```

configuration

```
get configuration
```

configuration Displays the configuration details for the policy in whose context you issue the **get configuration** command.

count

```
set policy { ... } [ count [ alarm { id_num1 id_num2 } ] ] { ... }
```

count Maintains a count, in bytes, of all the network traffic the policy allows to pass through the NetScreen device.

The **alarm** *number1 number2* parameter enables the alarm feature so that you can view alarms. You must enter the number of bytes per second (*number1*) and the number of bytes per minute (*number2*) required to trigger an alarm.

Example: The following command permits any kind of traffic from any address in the Trust zone to any address in the Untrust zone and maintains a count of all network traffic to which the policy applies:

```
set policy from trust to untrust any any any permit count
```

default-permit-all

```
set policy default-permit-all
```

default-permit-all Allows access without checking the access control list (ACL) for a matching policy.

disable

```
set policy [ global ] id pol_num disable
```

disable Disables the policy without removing it from the configuration.

di-alert-disable (within a policy context)

```
set di-alert-disable
```

di-alert-disable Disables DI (Deep Inspection) alert generation.

di-severity (within a policy context)

```
set di-severity
```

disable Specifies the severity of events that generate error messages. The possible event levels are **info**, **low**, **medium**, **high**, and **critical**.

from ... to

```
set policy { ... } from zone1 to zone2 src_addr dst_addr svc_name { ... } [ ... ]
```

from zone1 to zone2
src_addr dst_addr
svc_name

Specifies two zones between which a policy controls traffic.

- *zone1* is the name of the source security zone.
- *zone2* is the name of the destination security zone.
- *src_addr* is the name of the source address. Specifying **any** allows all source IP addresses.
- *dst_addr* is the name of the destination address. Specifying **any** allows all destination IP addresses.
- *svc_name* is the name of the service. Specifying **any** identifies all available services.

For more information on zones, see ["Zone Names" on page B-1](#).

Example: The following command permits HTTP traffic from any address in the Trust zone to any address in the Untrust zone:

```
set policy from trust to untrust any any HTTP permit
```

global

```
set policy global before { ... }  
set policy global id pol_num disable  
set policy global move pol_num1 { before pol_num2 | after pol_num3 }  
set policy global name name_str { ... }  
set policy global top
```

global

Creates or displays policies that use the Global zone. The The Global zone address book keeps all the VIPs of all interfaces, regardless of the zone to which the interface belongs. You can use these VIP addresses as destination addresses in policies between any two security zones.

id

```
get policy [ global ] id pol_num
    set policy [ global ] id pol_num1 { ... }
unset policy id pol_num [ disable ]
```

id *pol_num* Specifies an policy ID number. (The **disable** switch disables the policy.)

Example: The following command assigns the policy an ID value of 10 and permits FTP-GET traffic from any address in the Trust zone to any address in the Untrust zone:

```
set policy id 10 from trust to untrust any any ftp-get permit
```

l2tp

```
set policy [ global ] { ... } tunnel l2tp tunn_str { ... }
set policy [ global ] { ... } tunnel vpn tunn_str l2tp tunn_str
```

Example: The following command defines an inbound policy for an L2TP tunnel.

- VPN tunnel named "home2office"
- L2TP tunnel named "home-office"
- Dialup VPN group named "home_office"

```
set policy from untrust to trust dialup_vpn our_side any tunnel vpn home2office
    l2tp home_office
```

l2tp Specifies a Layer 2 Tunneling Protocol (L2TP) tunnel.

log

```
set policy [ global ] { ... } log [ alert ] { ... }
```

log [alert] Maintains a log of all connections to which the policy applies. The **alert** switch enables the syslog alert feature.

Example: The following command creates a policy and directs the NetScreen device to log the traffic to which the policy applies.

- Permits HTTP traffic from any address in the Trust zone to any address in the Untrust zone
- Directs the NetScreen device to log the traffic to which the policy applies
- Enables the syslog alert feature

```
set policy from trust to untrust any any HTTP permit log alert
```

move

```
set policy [ global ] move pol_num1 { before pol_num2 | after pol_num3 }
```

move Repositions a policy (*pol_num1*) before another policy (*pol_num2*) or after a policy (**pol_num3**) in the access control list (ACL). When one policy comes before another policy in the ACL, it has higher precedence.

Example: The following command positions a global policy with ID number 4 before the policy with ID number 2:

```
set policy global move 4 before 2
```

name

```
set policy [ global ] [ ... ] name name_str { ... }
```

name name_str Identifies the policy by name. (Assigning a name to an policy is optional.)

Example: The following command creates a new policy named "outbound":

```
set policy name outbound from trust to untrust any any any permit
```

nat

```
set policy [ global ] { ... } nat src [ dip-id id_num ] { ... }  
set policy [ global ] { ... } nat dst ip addr1 [ addr2 | port port_num ] { ... }
```

- nat** Enables or disables source and destination Network Address Translation (NAT-src and NAT-dst). This feature translates the original source or destination IP address in an IP packet header to another address.
- **src** Performs NAT-src on traffic to which the policy applies. The NetScreen device can perform NAT-src using the egress interface IP address (in which case, you do not specify a DIP pool) or with addresses from a Dynamic IP (DIP) pool:
 - **dip-id id_num** Specifies the ID number of a DIP pool. This number can be between 4 and 255.
 - **dst** Performs NAT-dst on traffic to which the policy applies. NetScreen supports the following three options for NAT-dst:
 - **ip addr1** Translates the original destination address to the address specified in the policy. The NetScreen device does not translate the original port number.
 - **ip addr1 addr2** Translates the original destination IP address from one range of addresses to an address in another range of addresses. The NetScreen device maintains a consistent mapping of an original destination address to a translated address within the specified range using a technique called address shifting.
 - **ip addr1 port port_num** Translates the original destination address and port number to the address and port number specified in the policy.

Example: The following command creates a policy that applies NAT-src on all traffic from any address in the Trust zone to any address in the Untrust zone and specifies DIP pool 8:

```
set policy from trust to untrust any any any nat src dip-id 8 permit
```

Example: The following commands create an address (1.1.1.5/32) named v-addr1 in the DMZ zone and a policy that applies NAT-dst on HTTP traffic from any address in the Untrust zone to the virtual destination address v-addr1 in the DMZ zone. The NetScreen device translates the destination address from 1.1.1.5 to 10.2.2.5:

```
set address dmz v-addr1 1.1.1.5/32  
set policy from untrust to dmz any v-addr1 http nat dst ip 10.2.2.5 permit
```

Example: The following command combines NAT src (source) and dst (destination):

```
set policy from trust to untrust any any any nat src dip-id 8 dst ip 10.2.2.5
  permit
```

negate

```
set { dst-address | src-address } negate
```

negate Applies the policy in whose context you issue this command to all addresses except those specified as either the destination (**dst-address**) or source (**src-address**). The negate option takes effect at the policy component level, applying to all items in the negated component

Example: The following commands permit HTTP traffic to the Untrust zone from all addresses in the Trust zone except from addr1:

```
ns-> set policy id 1 from trust to untrust any any http permit
ns-> set policy id 2 from trust to untrust addr1 any http permit
ns-> set policy id 2
ns(policy:2)-> set src-address negate
```

no-session-backup

```
set policy [ global ] { ... } no-session-backup { ... }
```

no-session-backup Disables backing up the sessions to which the policy applies when the NetScreen device is in a high availability (HA) configuration. By default, a NetScreen device operating in HA backs up sessions.

pair-policy

```
set policy [ global ] { ... } pair-policy pol_num [ ... ]
```

pair-policy *pol_num* Links the policy that you are configuring with another policy that references the same VPN tunnel so that both policies share one proxy ID and one security association (SA). This is useful when you want to allow bidirectional traffic over a policy-based VPN and there is source destination address translation using a DIP pool or destination address translation using a MIP or VIP. Without policy pairing, the NetScreen device derives a different proxy ID from both the outbound and inbound policies. This causes a problem for the remote peer if it has only a single proxy ID for the VPN tunnel. By pairing both policies together, they share a single proxy ID (derived from the policy that you configured last), which solves the proxy ID problem for the remote peer, and they share a single SA, which conserves SA resources.

Example: The following commands create two policies sharing the same VPN tunnel and then bind them into a policy pair. (You have previously created on the tunnel interface subnet a DIP pool with ID 4 and addresses 1.1.1.10 – 1.1.1.20, and a MIP from 1.1.1.5 to host 10.1.1.5.):

```
set policy id 1 from trust to untrust addr1 addr2 any nat src dip-id 4 tunnel vpn  
    vpn1  
set policy id 2 from untrust to trust addr2 mip(1.1.1.5) MAIL tunnel vpn vpn1  
    pair-policy 1
```

The proxy ID for both of these policies is as follows:

```
local 1.1.1.5/255.255.255.255, remote 10.2.2.0/255.255.255.0, proto 6, port 25
```

Because the local address in the above proxy ID does not include the addresses in the DIP pool or any service other than SMTP (or “MAIL”), you must also set a proxy ID with an address range that encompasses both the MIP (1.1.1.5) and DIP pool (1.1.1.10–1.1.1.20) and change the service to “ANY”:

```
set vpn vpn1 proxy-id local-ip 1.1.1.0/24 remote-ip 10.2.2.0/24 ANY
```

permit | deny

```
set policy [ global ] { ... } permit | deny [ ... ]
```

permit | deny

- **permit** allows the specified service to pass from the source address across the firewall to the destination address.
- **deny** blocks the service at the firewall.

Example: The following command:

- Defines a policy from the Trust zone to the Untrust zone
- Uses any source or destination IP address
- Permits any kind of service

```
set policy from trust to untrust any any any permit
```

schedule

```
set policy [ global ] { ... } schedule name_str [ ... ]
```

schedule

Applies the policy only at times defined in the specified schedule.

Example: With following commands, you first create a schedule named "Mkt_Sched" and then reference it in a policy permitting any kind of traffic from any address in the Trust zone to any address in the Untrust zone:

```
set schedule Mkt_Sched recurrent monday start 09:00 stop 12:00
set policy from trust to untrust any any any permit schedule Mkt_Sched
```

top

```
set policy [ global ] [ ... ] top
```

top Places the policy at the top of the access control list (ACL). The policy at the top of the ACL has the highest precedence.

Example: The following command:

- Permits any kind of service from any address in the Trust zone to any address in the Untrust zone
- Assigns to the policy an ID value of 30
- Places the policy at the top of the ACL

```
set policy id 30 top from trust to untrust any any any permit
```

traffic gbw

```
set policy [ global ] [ ... ] traffic  
    gbw number priority number mbw [ number ] dscp { disable | enable }
```

traffic gbw Defines the guaranteed bandwidth (GBW) in kilobits per second. The NetScreen device passes traffic below this threshold with the highest priority, without performing traffic shaping.

- **priority *number*** Specifies one of the eight traffic priority levels. When traffic falls between the guaranteed and maximum bandwidth settings, the NetScreen device passes traffic with higher priority first. Lower priority traffic is passed only if there is no higher priority traffic.
- **mbw *number*** Defines the maximum bandwidth (MBW) in kilobits per second. Traffic beyond this limit is throttled and dropped.
- **dscp { enable | disable }** Enables or disables a mapping of the eight NetScreen priority levels to the Differentiated Services—DiffServ—Codepoint (DSCP) marking system. In the NetScreen system, 0 is the highest priority and seven is the lowest.

Example: The following command:

- Permits HTTP traffic from any address in the Trust zone to any address in the Untrust zone
- Guarantees bandwidth of 3,000 kilobits per second
- Assigns a priority value of 2
- Sets the maximum bandwidth to 10,000 kilobits per second
- Enables mapping of the eight NetScreen priority levels to the DiffServ Codepoint (DSCP) marking system

```
set policy from trust to untrust any any HTTP permit traffic gbw 3000 priority 2
    mbw 10000 dscp enable
```

tunnel

```
set policy [ global ] { ... } tunnel
    { l2tp tunn_str | vpn-group id_num }
set policy [ global ] { ... } tunnel vpn tunn_str
    [ l2tp tunn_str | pair-policy pol_num ]
```

tunnel

Encrypts outgoing IP packets, and decrypts incoming IP packets.

- **vpn** [**l2tp** *tunn_str*] Identifies a VPN tunnel. For an IPSec VPN tunnel, specify **vpn** and the name of the VPN tunnel. For L2TP, specify **vpn** (with the name of the VPN tunnel) and **l2tp** (with the name of the L2TP tunnel).
- **vpn** [**pair-policy** *id_num*] Links this policy with an existing policy also referencing the same VPN. The VPN uses the proxy-id derived from the policy whose configuration includes the **pair-policy** keyword.
- **vpn-group** *id_num* Identifies a VPN group (*id_num*). A VPN group consist of multiple VPNs, which you can specify in a single policy.
- **vpn-tunnel** Identifies an active tunnel.

Example: The following command defines a policy that uses a defined VPN tunnel.

- Encrypts traffic exchanged with the corporate headquarters (denoted by address book entry Headquarters)
- Uses a VPN named To_HQ:

```
set policy from trust to untrust any Headquarters any tunnel vpn To_HQ
```

verify

```
exec policy verify [ from zone [ to zone ] | global | to zone ]
```

verify

Verifies that the order of policies in a policy list is valid so that a policy higher in the list does not eclipse, or "shadow", another policy lower in the list. If the verification check discovers policy shadowing, the command output explains which policies are shadowing which. You can define the scope of the verification as follows:

- Not setting any further options instructs the NetScreen device to verify the ordering of policies in all policy sets.
- **from zone** Checks the ordering of policies from the specified zone to any zone.
- **from zone to zone** Checks the ordering of policies between the specified zones.
- **global** Checks the ordering of policies in the global policy set.
- **to zone** Checks the ordering of policies from any zone to the specified zone.

Example: The following command verifies the ordering of policies from the Trust zone to the Untrust zone:

```
exec policy verify from trust to untrust
```


port-mode

Description: Use the **port-mode** commands to set the port, interface, and zone bindings for the NetScreen-5XT and NetScreen-5GT devices. (Use the **get system** command to see the current port mode setting.)

Warning: Setting the port mode removes any existing configurations on the device and requires a system reset.

Syntax

exec

```
exec port-mode { trust-untrust | home-work | dual-untrust | combined }
```

Keywords and Variables

trust-untrust

Defines the following port, interface, and zone bindings:

- Binds the Untrusted Ethernet port to the untrust interface, which is bound to the Untrust zone.
- Binds the Trusted1 through Trusted4 Ethernet ports to the trust interface, which is bound to the Trust zone.
- Binds the Modem port to the serial interface, which you can bind as a backup interface to the Untrust zone.

This is the default port mode.

home-work

Defines the following port, interface, and zone bindings:

- Binds the Untrusted Ethernet port to the ethernet3 interface, which is bound to the Untrust zone.
- Binds the Trusted4 and Trusted3 Ethernet ports to the ethernet2 interface, which is bound to the Home zone.
- Binds the Trusted2 and Trusted1 Ethernet ports to the ethernet1 interface, which is bound to the Work zone.
- Binds the Modem port to the serial interface, which you can bind as a backup interface to the Untrust zone.

dual-untrust

Defines the following port, interface, and zone bindings:

- Binds the Untrusted port to the ethernet3 interface, which is bound to the Untrust zone.
- Binds the Trusted4 Ethernet port to the ethernet2 interface, which is bound as a backup interface to the Untrust zone.
- Binds the Trusted1 through Trusted3 Ethernet ports to the ethernet1 interface, which is bound to the Trust zone.

combined

Defines the following port, interface, and zone bindings:

- Binds the Untrusted Ethernet port to the ethernet4 interface, which is bound to the Untrust zone.
- Binds the Trusted4 Ethernet port to the ethernet3 interface, which is bound as a backup interface to the Untrust zone.
- Binds the Trusted3 and Trusted2 Ethernet ports to the ethernet2 interface, which is bound to the Home zone.
- Binds the Trusted1 Ethernet port to the ethernet1 interface, which is bound to the Work zone.

The combined port mode is supported only on the NetScreen-5XT Elite (unrestricted users) platform.

Warning: Setting the port mode removes any existing configurations on the device and requires a system reset.

“pppoe” through “zone”

This section lists and describes Juniper Networks NetScreen Command Line Interface (CLI) commands **pppoe** through **zone**.

Note: As you execute CLI commands using the syntax descriptions in this section, you may find that certain commands and command features are unavailable on your NetScreen platform. A good example is the **vsys** command, which is available on a NetScreen-500 device, but not on a NetScreen-5XP device. Similarly, some command options are unavailable on certain platforms, as with the **df-bit** option of the **vpn** command. This option is available on a NetScreen-500, but not on a NetScreen-5XP.

pppoe

Description: Use the **pppoe** commands to configure PPPoE, or to display current PPPoE configuration parameters.

Point-to-Point Protocol over Ethernet (PPPoE) is a protocol that allows the members of an Ethernet LAN to make individual PPP connections with their ISP by encapsulating the IP packet within the PPP payload, which is encapsulated inside the PPPoE payload. Some NetScreen devices support PPPoE, which allows them to operate compatibly on DSL, Ethernet Direct, and cable networks run by ISPs that use PPPoE to give their clients Internet access.

Syntax

clear

```
clear [ cluster ] pppoe [ name name_str ]
```

exec

```
exec pppoe [ name name_str ] { connect | disconnect }
```

get

```
get pppoe  
[  
  all |  
  name name_str |  
  id id_num  
]  
[ configuration | statistics ]
```

set

```
set pppoe [ name name_str ]
{
  ac name_str |
  authentication { CHAP | PAP | any } |
  auto-connect number |
  clear-on-disconnect |
  idle-interval number |
  interface [ name_str ] |
  netmask mask |
  ppp
  {
    lcp-echo-retries number |
    lcp-echo-timeout number
  } |
  service name_str |
  static-ip |
  update-dhcpserver |
  username name_str password pswd_str
}
```

unset

```
unset pppoe [ name name_str ]
{
  ac |
  authentication { CHAP | PAP } |
  auto-connect |
  clear-on-disconnect |
  idle-interval |
  interface |
  netmask |
  ppp
  {

```

```
        lcp-echo-retries |
        lcp-echo-timeout
    } |
    service |
    static-ip |
    update-dhcpserver |
    username
}
```

Keywords and Variables

ac

```
set pppoe ac name_str
unset pppoe ac
```

ac Allows the interface to connect only to the specified AC (access concentrator).

all

```
get pppoe all
```

all Displays information for all PPPoE instances.

authentication

```
set pppoe authentication { CHAP | PAP | any }
unset pppoe authentication { CHAP | PAP }
```

authentication Sets the authentication methods to **CHAP**, **PAP**, or **any**. (The **any** option gives preference to CHAP.) The default of authentication is any (both CHAP and PAP). To set authentication to CHAP only, first execute **unset pppoe authentication PAP**.

auto-connect

```
set pppoe auto-connect number  
unset pppoe auto-connect
```

auto-connect Specifies the number of seconds that elapse before automatic re-initiation of a previously-closed connection occurs. Valid range is 0-10000. (0 to disable.)

clear-on-disconnect

```
set pppoe [ name name_str ] clear-on-disconnect  
unset pppoe clear-on-disconnect
```

clear-on-disconnect Directs the NetScreen device to clear the IP address and the gateway for the interface once PPPoE disconnects. By default, this is disabled; that is, the IP address and gateway for the interface remain when PPPoE disconnects.
If you do not specify **name**, ScreenOS sets the parameter for the default instance untrust.

cluster

```
clear cluster pppoe
```

cluster Propagates the **clear** operation to all other devices in a NSRP cluster.

configuration

```
get pppoe [ name name_str ] configuration
```

configuration Displays the configuration options.
If you do not specify **name**, ScreenOS displays the parameters for the default instance untrust.

connect | disconnect

```
exec pppoe [ name name_str ] { connect | disconnect }
```

connect Starts a PPPoE connection for an instance. (Each instance can be bound to an interface.)

disconnect Takes down a PPPoE connection.

idle-interval

```
set pppoe idle-interval number
```

```
unset pppoe idle-interval
```

idle-interval Sets the idle timeout, which is time elapsed (in minutes) before the NetScreen device terminates a PPPoE connection due to inactivity. Specifying 0 turns off the idle timeout and the device never terminates the connection.

id

```
get pppoe id id_num
```

id Specifies a PPPoE instance by ID number.

interface

```
set pppoe interface [ name_str ]
```

```
unset pppoe interface
```

interface Specifies the interface for PPPoE encapsulation.

name

```
exec pppoe [ name name_str ] { connect | disconnect }
get pppoe [ name name_str | all ]
set pppoe [ name name_str ] ...
unset pppoe [ name name_str ]
```

name Specifies or defines the name for a specific PPPoE instance. You can assign a username and password, interface, and other PPP/PPPoE parameters to the instance.
If you do not specify **name**, ScreenOS automatically configures the parameters for the default instance untrust.

Example: The following commands define a name for a PPPoE instance.

- User name user1 and password 123456
- PPPoE instance pppoe-user-1 bound to the ethernet2 interface

```
set pppoe name pppoe-user-1 username user1 password 123456
set pppoe name pppoe-user-1 interface ethernet2
```

netmask

```
set pppoe netmask mask
unset pppoe netmask
```

netmask Specifies a PPPoE subnet mask that the device assigns to the interface bound to the PPPoE instance (after establishment of the connection).
When it is necessary for two or more interfaces to have overlapping subnets, use the following command:
set vrouter vrouter ignore-subnet-conflict

ppp

```
set pppoe ppp { ... }  
unset pppoe ppp { ... }
```

ppp Specifies PPP parameters.

- **lcp-echo-retries** the number of unacknowledged Lcp Echo requests before connection is terminated. Valid range is 1-30.
- **lcp-echo-timeout** the time that elapses between transmission of two Lcp Echo requests. Valid range is 1-1000 seconds.

service

```
set pppoe service name_str  
unset pppoe service
```

service Allows only the specified service (*name_str*).

static-ip

```
set pppoe static-ip  
unset pppoe static-ip
```

static-ip Specifies that your connection uses the IP address assigned to your device's interface.

statistics

```
get pppoe statistics
```

statistics Specifies the statistics information.

update-dhcpserver

```
set pppoe update-dhcpserver
unset pppoe update-dhcpserver
```

update-dhcpserver Specifies that the DHCP server (on the device) automatically updates DNS parameters received through the PPPoE connection.

user-name

```
set pppoe username name_str password pswd_str
```

username Sets the user name and password.

Example: The following command sets the username to "Phred", and Phred's password to "!@%)&&":

```
set pppoe username Phred password !@%)&&
```

Defaults

The defaults for this command are as follows.

- Disabled
- Authentication method **any**
- Timeout 30 minutes
- **auto-connect** setting disabled
- **lcp-echo-timeout** value 180 seconds
- **retries** value 10
- **netmask** value 255.255.255.255
- **update-dhcpserver** setting enabled
- **static-ip** setting disabled
- **clear-on-disconnect** setting disabled

proxy-id

Description: Use the **proxy-id** commands to set device behavior for processing proxy ID updates. A proxy ID is a three-part tuple consisting of local IP address, remote IP address, and service. The proxy ID for both peers must match, which means that the service specified in the proxy ID for both peers must be the same, and the local IP address specified for one peer must be the same as the remote IP address specified for the other peer. The peers exchange proxy IDs during IKE Phase 2 negotiations.

During the startup process, the NetScreen device loads its configuration file. While loading this file, the NetScreen device reads the policies before the routes. Because of this, routing information that involves MIPs or VIPs can result in the NetScreen device deriving incorrect proxy-IDs from the policy information in the file. To resolve this problem, you can use the **unset proxy-id manual-update** command to change the default behavior of the device to update proxy IDs after the configuration file finishes loading. However, if you have a large number of policies, the update procedure can take a very long time to complete. By default, the device behavior does not update proxy IDs automatically during startup. Instead, you must manually update proxy IDs by entering the **exec proxy-id update** command. For VPN traffic that uses source or destination address translation, Juniper Networks recommends either of the following approaches:

- Use routing-based VPNs and separate the VPN and its manually defined proxy ID from the policy that enforces address translation.
- Use policy-based VPNs and assign proxy IDs to the VPN tunnels referenced by the policies rather than allow the NetScreen device to automatically derive the proxy IDs from the policies.

Syntax

exec

exec proxy-id update

get

get proxy-id

set

```
set proxy-id manual-update
```

unset

```
unset proxy-id manual-update
```

Keywords and Variables

update

```
exec proxy-id update
```

update Instructs the NetScreen device to update all VPN proxy IDs.

manual-update

```
set proxy-id manual-update  
unset proxy-id manual-update
```

manual-update When set, instructs the NetScreen device to update all VPN proxy IDs only in response to the **exec proxy-id update** command. When unset, instructs the NetScreen device to update the proxy IDs automatically during route change.

Defaults

By default, the NetScreen device does not update proxy IDs automatically.

reset

Description: Use the **reset** command to restart the NetScreen device.

Syntax

```
reset
[
  no-prompt |
  save-config { no | yes } [ no-prompt ]
]
```

Keywords and Variables

no-prompt

```
reset no-prompt
```

no-prompt Indicates no confirmation.

save-config

```
reset save-config [ no | yes ] [ no-prompt ]
```

- save-config**
- **no** Directs the NetScreen device to not save the current configuration before resetting.
 - **yes** Directs the NetScreen device to save the current configuration before resetting.
 - **no-prompt** Does not display a confirmation prompt.

RIP Commands

Description: Use the **rip** context to begin configuring Routing Information Protocol (RIP) for a NetScreen virtual router.

Context Initiation

Initiating the **rip** context can take up to four steps.

1. Enter the vrouter context by executing the **set vrouter** command.

```
set vrouter vrouter
```

For example:

```
set vrouter trust-vr
```

2. Enter the **rip** context by executing the **set protocol rip** command.

```
ns(trust-vr)-> set protocol rip
```

3. Enable RIP (it is disabled by default).

```
ns(trust-vr/rip)-> set enable
```

RIP Commands

The following commands are executable in the **rip** context.

advertise-def-route

Use the **advertise-def-route** commands to advertise the default route (0.0.0.0/0) of the current virtual router the RIP routing domain.

Every virtual router may have a default route entry, which matches every destination. (Any entry with a more specific prefix overrides the default route entry.)

Command options: **get, set, unset**

config

Use the **config** command to display all commands executed to configure the RIP routing instance.

Command options: **get**

default-metric	Use the default-metric commands to set the RIP metric for redistributed routes. The default value is 10. Command options: set, unset
enable	Use the enable commands to enable or disable RIP in the virtual router. Command options: set, unset
flush-timer	Use the flush-timer commands to configure the number of seconds that elapse before ScreenOS automatically removes an invalidated route. The default is 120 seconds. Command options: set, unset
garbage-list	Displays all routes currently contained in the RIP garbage list. This list contains routes automatically removed from the routing table because the device did not obtain the routes in the time interval specified by the Invalid Timer setting. When the Flush Timer interval elapses for an entry, the device purges the entry from the garbage list. Command options: get
interface	Use the interface command to display all RIP interfaces in the virtual router. Command options: get
invalid-timer	Use the invalid-timer commands to configure the number of seconds that elapse after a neighbor stops advertising a route before the route becomes invalid. The default is 180 seconds. Command options: set, unset
max-neighbor-count	Use the max-neighbor-count commands to set the maximum number of RIP neighbors allowed. The default is 16. Command options: set, unset
neighbors	Use the neighbors command to display the status of RIP neighbors. Command options: get
no-source-validation	Use the no-source-validation commands to accept responses from RIP neighbors in other subnets or to reject such responses. The default action is to reject the responses. Command options: set, unset
redistribute	Use the redistribute commands to import known routes from a router running a different protocol into the current routing instance.

You can import the following types of routes:

- Manually created (static) routes
- BGP routes
- OSPF routes
- Routes created by an external router, due to reconnection of an interface with an IP address
- Routes imported from other virtual routes

Command options: **set, unset**

reject-default-route

Use the **reject-default-route** commands to cause RIP to reject a default route learned from another protocol.

Command options: **get, set, unset**

route-map

Use the **route-map** commands to filter routes and offset the metric to a RIP route matrix.

Command options: **get, set, unset**

routes-redistribute

Use the **routes-redistribute** command to display redistributed routes.

Command options: **get**

rules-redistribute

Use the **rules-redistribute** command to display redistribution rules.

Command options: **get**

threshold-update

Use the **threshold-update** commands to set the maximum number of routing packets allowed per update interval.

Command options: **set, unset**

timer

Use the **timer** command to display RIP timers.

Command options: **get**

trusted-neighbors

Use the **trusted-neighbors** commands to set an access list that defines RIP neighbors.

Command options: **get, set, unset**

update-timer

Use the **update-timer** commands to set the interval, in seconds, when route updates are issued to RIP neighbors.

Command options: **set, unset**

update-threshold

Use the **update-threshold** command to display the number of routing packets per update interval.

Command options: **get**

advertise-def-route

Description: Use the **advertise-def-route** commands to advertise the default route (0.0.0.0/0) of the current virtual router.

Every router might have a default route entry, which matches every destination. (Any entry with a more specific prefix overrides the default route entry.)

Before you can execute the **advertise-def-route** commands, you must initiate the **rip** context. (See [“Context Initiation” on page 409.](#))

Syntax

get

```
get advertise-def-route
```

set

```
set advertise-def-route [ always ] {metric number | preserve-metric }
```

unset

```
unset advertise-def-route
```

Keywords and Variables

always

```
set advertise-def-route always ...
```

always Directs the routing instance to advertise the default route under all conditions, even if there is no default route in the routing table. If you specify **always**, you must also specify the **metric** parameter; you can optionally specify the **preserve-metric** parameter. If you do not specify **always**, you must specify either the **metric** or **preserve-metric** option.

metric

```
set advertise-def-route always metric number
```

metric Specifies the metric (cost), which indicates the overhead associated with the default route. Enter a number between 1 and 15. You must specify this parameter if you specify the **always** option.

preserve-metric

```
set advertise-def-route ... [ preserve-metric ]
```

preserve-metric Instructs the virtual router to use the original (source) route's metric for advertisement when the route is redistributed. When you execute a **preserve-metric** command, in conjunction with a value specified by the **metric** command, the **preserve-metric** parameter takes precedence over the metric value when a route is redistributed.

config

Description: Use the **config** command to display all commands executed to configure the RIP local virtual router. Before you can execute the **config** command, you must initiate the **rip** context. (See ["Context Initiation" on page 409.](#))

Syntax

get

get config

Keywords and Variables

None.

default-metric

Description: Use the **default-metric** commands to set the RIP metric for redistributed routes.

Before you can execute the **default-metric** commands, you must initiate the **rip** context. (See ["Context Initiation" on page 409.](#))

Syntax

set

```
set default-metric number
```

unset

```
unset default-metric
```

Keywords and Variables

Variable Parameter

```
set default-metric number
```

number

The metric for the routes redistributed into RIP. This metric value can be from 1 to 15.

enable

Description: Use the **enable** commands to enable or disable RIP from the current virtual router.

Before you can execute the **enable** commands, you must initiate the **rip** context. (See [“Context Initiation” on page 409.](#))

Syntax

set

set enable

unset

unset enable

Keywords and Variables

None.

flush-timer

Description: Use the **flush-timer** commands to configure the time that elapses before an invalid route is removed. Before you can execute the **flush-timer** commands, you must initiate the **rip** context. (See [“Context Initiation” on page 409.](#))

Syntax

set

set flush-timer *number*

unset

unset flush-timer

Keywords and Variables

Variable Parameter

set flush-timer *number*

number

The number of seconds that elapses before an invalid route is removed. This value must be greater than the current **update-timer** value. The default value is 120 seconds.

garbage-list

Description: Use the **garbage-list** commands to display all routes currently contained in the RIP garbage list. The garbage list contains routes automatically removed from the routing table because the device did not obtain the routes in the time interval specified by the Invalid Timer setting. When the Flush Timer interval elapses for an entry, the device automatically purges the entry from the garbage list.

Before you can execute the **garbage-list** commands, you must initiate the **rip** context. (See [“Context Initiation” on page 409.](#))

Syntax

get

get garbage-list

Keywords and Variables

None.

interface

Description: Use the **interface** command to display all RIP interfaces on the current virtual router.

Before you can execute the **interface** command, you must initiate the **rip** context. (See [“Context Initiation” on page 409.](#))

Syntax

get

get interface

Keywords and Variables

None.

invalid-timer

Description: Use the **invalid-timer** commands to configure the time that elapses after a neighbor stops advertising a route before the route becomes invalid.

Before you can execute the **invalid-timer** commands, you must initiate the **rip** context. (See [“Context Initiation” on page 409.](#))

Syntax

set

```
set invalid-timer number
```

unset

```
unset invalid-timer
```

Keywords and Variables

Variable Parameter

```
set invalid-timer number
```

number

The number of seconds after a neighbor stops advertising a route that the route becomes invalid. This value must be greater than the current **update-timer** value. The default value is 180 seconds.

max-neighbor-count

Description: Use the **max-neighbor-count** commands to set the maximum number of RIP neighbors allowed. Before you can execute the **max-neighbor-count** commands, you must initiate the **rip** context. (See [“Context Initiation” on page 409.](#))

Syntax

set

set max-neighbor-count *number*

unset

unset max-neighbor-count

Keywords and Variables

Variable Parameter

set max-neighbor-count *number*

number

The maximum number of RIP neighbors allowed. This value can be from one to the maximum value possible for your NetScreen device. The default is 16 RIP neighbors.

neighbors

Description: Use the **neighbors** command to display the status of all RIP neighbors.

Before you can execute the **neighbors** command, you must initiate the **rip** context. (See [“Context Initiation” on page 409.](#))

Syntax

get

get neighbors

Keywords and Variables

None.

no-source-validation

Description: Use the **no-source-validation** commands to accept responses from RIP neighbors in different subnets. If you do not set this switch, the virtual router does not process responses from neighbors in other subnets.

Before you can execute the **no-source-validation** commands, you must initiate the **rip** context. (See [“Context Initiation” on page 409.](#))

Syntax

set

set no-source-validation

unset

unset no-source-validation

Keywords and Variables

None.

redistribute

Description: Use the **redistribute** commands to import known routes from a router running a different protocol into the current RIP routing instance.

You can import the following types of routes:

- Manually-created routes (**static**)
- BGP routes (**bgp**)
- OSPF routes (**ospf**)
- Directly-connected interface with an IP address assigned to it (**connected**)
- Routes that have already been imported (**imported**)

Before you can execute the **redistribute** commands, you must initiate the **rip** context. (See [“Context Initiation” on page 409.](#))

Syntax

get

```
get routes-redistribute
get rules-redistribute
```

set

```
set redistribute route-map name_str protocol
    { bgp | connected | imported | ospf | static }
```

unset

```
unset redistribute route-map name_str protocol
    { bgp | connected | imported | ospf | static }
```

Keywords and Variables

protocol

```
set redistribute route-map name_str protocol { ... }
```

protocol

Specifies the routing protocol type. The route map can use the protocol type to determine whether to permit or deny a route.

- **bgp** specifies that the route map performs an action only on BGP routes in the subnetwork.
- **connected** specifies that the route map performs an action only on routes sent from an external router that has at least one interface with an IP address assigned to it.
- **imported** specifies that the route map performs an action only on imported routes in the subnetwork.
- **ospf** specifies that the route map performs an action only on OSPF routes in the subnetwork.
- **static** specifies that the route map performs an action only on static routes in the subnetwork.

route-map

```
set redistribute route-map name_str protocol { ... }
```

route-map

Identifies the route map that specifies the routes to be imported.

Example: The following command redistributes a route that originated from a BGP routing domain into the current RIP routing instance:

```
ns(trust-vr/rip)-> set redistribute route-map map1 protocol bgp
```

reject-default-route

Description: Use the **reject-default-route** commands to cause RIP to reject default routes learned from a neighbor in the RIP domain.

Before you can execute the **reject-default-route** commands, you must initiate the **rip** context. (See [“Context Initiation” on page 409.](#))

Syntax

get

`get reject-default-route`

set

`set reject-default-route`

unset

`unset reject-default-route`

Keywords and Variables

None.

route-map

Description: Use the **route-map** commands to filter incoming or outgoing routes.

Before you can execute the **route-map** commands, you must initiate the **rip** context. (See [“Context Initiation” on page 409.](#))

Syntax

get

```
get route-map
```

set

```
set route-map name_str { in | out }
```

unset

```
set route-map name_str { in | out }
```

Keywords and Variables

Variable Parameter

```
set route-map name_str
```

name_str The name of the route map to filter routes.

in

```
set route-map name_str in
```

in Specifies the route map is applied to routes to be learned by RIP.

out

set route-map *name_str* **out**

out Specifies the route map is applied to routes to be advertised by RIP.

Example: The following command applies the route map map1 to routes to be advertised by RIP:

```
ns(trust-vr/rip)-> set route-map map1 out
```

routes-redistribute

Description: Use the **routes-redistribute** command to display details about routes imported from other protocols into RIP.

Before you can execute the **routes-redistribute** command, you must initiate the **rip** context. (See [“Context Initiation” on page 409.](#))

Syntax

get

get routes-redistribute

Keywords and Variables

None.

rules-redistribute

Description: Use the **rules-redistribute** command to display conditions set for routes imported from other protocols into RIP.

Before you can execute the **rules-redistribute** command, you must initiate the **rip** context. (See [“Context Initiation” on page 409.](#))

Syntax

get

`get rules-redistribute`

Keywords and Variables

None.

threshold-update

Description: Use the **threshold-update** commands to set the maximum number of routing packets received and processed per update interval, per neighbor.

Before you can execute the **threshold-update** commands, you must initiate the **rip** context. (See [“Context Initiation” on page 409.](#))

Syntax

set

set threshold-update *number*

unset

unset threshold-update

Keywords and Variables

Variable Parameter

set threshold-update *number*

<i>number</i>	The maximum number of routing packets allowed per update interval. This value must be greater than zero.
---------------	--

timer

Description: Use the **timer** command to display information about various RIP timers.

Before you can execute the **timer** command, you must initiate the **rip** context. (See ["Context Initiation" on page 409.](#))

Syntax

get

`get timer`

Keywords and Variables

None.

trusted-neighbors

Description: Use the **trusted-neighbors** commands to specify an access list that defines allowed RIP neighbors. Before you can execute the **trusted-neighbors** commands, you must initiate the **rip** context. (See [“Context Initiation” on page 409.](#))

Syntax

get

```
get trusted-neighbors
```

set

```
set trusted-neighbors id_num
```

unset

```
set trusted-neighbors id_num
```

Keywords and Variables

Variable Parameter

```
set trusted-neighbors id_num
```

id_num The number of the access list that defines the allowed RIP neighbors.

update-timer

Description: Use the **update-timer** commands to set the interval that RIP sends route updates to neighbors.

Before you can execute the **update-timer** commands, you must initiate the **rip** context. (See ["Context Initiation" on page 409.](#))

Syntax

set

set update-timer *number*

unset

unset update-timer

Keywords and Variables

Variable Parameter

set update-timer *number*

number

The interval, in seconds, that RIP sends route updates to neighbors. This value must be at least one, and no greater than the current **invalid-timer** value. The default is 30 seconds.

update-threshold

Description: Use the **update-threshold** command to display the number of routing packets per update interval.

Before you can execute the **update-threshold** command, you must initiate the **rip** context. (See ["Context Initiation" on page 409.](#))

Syntax

get

get update-threshold

Keywords and Variables

None.

route

Description: Use the **route** commands to display entries in the static route table.

The **get route** command displays:

- The IP address, netmask, interface, gateway, protocol, preference, metric, and owner vsys
- The **protocol** value can be any of the following:
 - **C** (Connected)
 - **S** (Static)
 - **A** (Auto Exported)
 - **I** (Imported; that is, route imported from another virtual router)
 - **iB** (internal BGP)
 - **eB** (external BGP)
 - **O** (OSPF)
 - **E1** (OSPF external type 1)
 - **E2** (OSPF external type 2)

Use the **get route** command to see if the NetScreen device has a route to the IP address on the correct interface.

Syntax

get

```
get route
[
  id id_num |
  ip [ ip_addr ] |
  prefix ip_addr/mask |
  protocol { bgp | connected | imported | ospf | rip | static } |
  summary
]
```

Keywords and Variables

id

```
get route id id_num
```

id Displays a specific route for the ID number *id_num*.

Example: The following command displays the route information for a route with ID number 477:

```
get route id 477
```

ip

```
get route ip ip_addr
```

ip Displays a specific route for the target IP address (*ip_addr*).

Example: The following command displays the route information to a machine with the IP address 172.16.60.1:

```
get route ip 172.16.60.1
```

prefix

```
get route prefix ip_addr/mask
```

prefix Displays routes within a specified subnet (*ip_addr/mask*).

Example: The following command displays the routes within the subnet 1.1.1.1/24:

```
get route prefix 1.1.1.1/24
```

protocol

```
get route protocol { bgp | connected | imported | ospf | rip | static }
```

protocol	Specifies the routing protocol, and directs the NetScreen device to display the routes derived from that protocol. <ul style="list-style-type: none">• bgp Directs the device to display BGP routes.• connected Directs the device to display only routes sent from an external router that has at least one interface with an IP address assigned to it.• imported Directs the device to display imported routes.• rip Directs the device to display RIP routes.• ospf Directs the device to display only OSPF routes.• static Directs the device to display only static routes.
-----------------	--

summary

```
get route summary
```

summary	Displays summary information, including number of routes, for each protocol.
----------------	--

Defaults

The **get route** command displays all entries in the route table unless a particular target IP address is specified.

Description: Use the **sa** commands to display active or inactive security associations (SAs) or to clear a specified SA.

A *security association* (SA) is a unidirectional agreement between VPN participants regarding the methods and parameters to use while securing a communication channel. Full bidirectional communication requires at least two SAs, one for each direction.

An SA groups together the following components for securing communications:

- Security algorithms and keys
- Protocol mode (transport or tunnel)
- Key management method (Manual Key or AutoKey IKE)
- SA lifetime

For outbound VPN traffic, a security policy invokes the SA associated with the VPN tunnel. For inbound traffic, the NetScreen device looks up the SA by using the following triplet: destination IP, security protocol (AH or ESP), and security parameter index (SPI) value, which are sent to the peer in the first message of a Phase 1 IKE exchange.

Syntax

clear

```
clear [ cluster ] sa id_num
```

get

```
get sa  
  [  
    id id_num |  
    [ active | inactive ] stat  
  ]
```

Keywords and Variables

Variable Parameter

clear [**cluster**] **sa** *id_num*

id_num Specifies the SA ID number.

cluster

clear cluster sa *id_num*

cluster Propagates the **clear** operation to all other devices in a NSRP cluster.

id

get sa id *id_num*

id Displays a specific IPSec Security Association (SA) entry with the ID number.

stat

get sa [...] **stat**

stat Shows the SA statistics for the device.
Displays these statistics for all incoming or outgoing SA pairs:

- **Fragment:** The total number of fragmented incoming and outgoing packets.
- **Auth-fail:** The total number of packets for which authentication has failed.
- **Other:** The total number of miscellaneous internal error conditions other than those listed in the auth-fail category.
- **Total Bytes:** The amount of active incoming and outgoing traffic

sa-statistics

Description: Use the **sa-statistics** command to clear all statistical information (such as the number of fragmentations and total bytes through the tunnel) in a security association (SA) for an AutoKey IKE VPN tunnel.

Syntax

clear

```
clear [ cluster ] sa-statistics [ id id_num ]
```

Keywords and Variables

cluster

```
clear cluster sa-statistics id id_num
```

cluster If the NetScreen device is in a high availability (HA) configuration, propagates the **clear** operation to all other devices in the NetScreen Redundancy Protocol (NSRP) cluster.

id

```
clear [ cluster ] sa-statistics id id_num
```

id id_num Clears the SA statistics for a particular SA.

Example: The following command clears the SA statistics for SA 2:

```
clear sa-statistics id 2
```

save

Description: Use the **save** commands to save the NetScreen device configuration settings either to flash memory, a flash memory card, or a Trivial File Transfer Protocol (TFTP) server.

Syntax

save

save

save config

```
save config
[
  all-virtual-system |
  to
  {
    flash |
    slot1 filename |
    tftp ip_addr filename
  }
  [ merge ] |
from
{
  flash |
  slot1 filename |
  tftp ip_addr filename
  [
    merge |
    to
    {
      flash [ from interface ] |
      last-known-good |
```



```
        slot1 filename |
        tftp ip_addr filename [ from interface ]
    }
    [ from interface ]
}
```

save image-key

```
save image-key tftp ip_addr filename from interface
```

save software

```
save software from
{
    flash |
    slot1 filename |
    tftp ip_addr filename
}
to
{
    flash |
    slot1 filename |
    tftp ip_addr filename
}
[ from interface ]
```

Keywords and Variables

all-virtual-system

save config all-virtual-system

all-virtual-system Saves all virtual system configurations.

flash

```
save config from { ... } to flash [ from interface ]
save config from flash to { ... } [ from interface ]
save software from flash to { ... } [ from interface ]
save software from { ... } flash to [ from interface ]
```

flash Saves from (or to) flash memory. The **from interface** option specifies the source interface if you specify TFTP.

Example: The following command saves the current configuration from flash memory to a file (output.txt) on a TFTP server (172.16.10.10):

```
save config from flash to tftp 172.16.10.10 output.txt
```

from { ... } to

```
save config from { ... } to { ... }
save software from { ... } to { ... }
```

from Saves from the specified source.

to Saves to the specified destination.

Example: The following command saves the current configuration from flash memory to a file (output.txt) on a TFTP server (IP address 172.16.10.10):

```
save config from flash to tftp 172.16.10.10 output.txt
```

last-known-good

```
save config to last-known-good
```

last-known-good Saves the current configuration to flash memory as the LKG (last-known-good) configuration. The NetScreen device can revert to this LKG file by doing a configuration rollback. The NetScreen device automatically names the LKG file *\$LKG\$.cfg*. You cannot rename the LKG file or give it a different name upon saving it.

merge

```
save config from { ... } merge [ from interface ]
```

merge Merges the saved configuration with the current configuration. The **from interface** option specifies the source interface.

Example: The following command merges the current configuration with the configuration in a file (input.txt) on a TFTP server (IP address 172.16.10.10):

```
save config from tftp 172.16.10.10 input.txt merge
```

slot1

```
save config from { ... } to slot1 [ ... ]
save config from slot1 to { ... }
save software from slot1 to { ... }
save software from { ... } to slot1 [ ... ]
```

slot1 Saves from (or to) a file in the memory card slot.

Example: The following commands saves the current configuration from a file (input.txt) in the the slot1 memory card to flash memory:

```
save config from slot1 input.txt to flash
```

tftp

```
save config from tftp filename to { ... } [ from interface ]  
save image-key tftp ip_addr filename from interface  
save software from tftp filename to { ... } [ from interface ]
```

tftp Saves from (or to) a file on a TFTP server.

Example: The following command loads an authentication key on a FIPS-compliant NetScreen device from a file named nskey.cer on a TFTP server at 10.10.1.2:

```
save image-key tftp 10.10.1.2 nskey.cer
```

scheduler

Description: Use the **scheduler** commands to create or modify a schedule, or to display the settings in a schedule.

A *schedule* is a configurable object that you can use to define when policies are in effect. NetScreen devices use schedules to enforce the policies at specified times or intervals. Through the application of schedules, you can control network traffic flow and enforce network security.

Syntax

get

```
get scheduler [ name name_str | once | recurrent ]
```

set

```
set scheduler name_str
[
  once start date time stop date time [ comment string ] |
  recurrent
    { monday | tuesday | wednesday | thursday | friday | saturday | sunday
      start time stop time
    }
    [ start time stop time ]
    [ comment string ]
]
```

unset

```
unset scheduler name_str
```

Keywords and Variables

name

```
get scheduler name name_str
```

name *name_str* Defines a name for the schedule.

once

```
get scheduler once
```

```
set scheduler name_str once start date time stop date time [ ... ]
```

once Apply the schedule once, starting on the day, month, year, hour, and minute defined, and stopping on the month, day, year, hour, and minute defined.

recurrent

```
get scheduler recurrent
```

```
set scheduler name_str recurrent { ... } [ ... ]
```

recurrent Directs the NetScreen device to repeat the schedule according to the defined day of the week, hour, and minutes.

- **monday** Repeats every Monday.
- **tuesday** Repeat every Tuesday.
- **wednesday** Repeat every Wednesday.
- **thursday** Repeat every Thursday.
- **friday** Repeat every Friday.
- **saturday** Repeat every Saturday.
- **sunday** Repeat every Sunday.

- **start** Defines when to start the schedule.
- **stop** Defines when to stop the schedule.
- **comment** Defines a descriptive character string.

start | stop

```
set scheduler name_str once start date time stop date time [ ... ]  
set scheduler name_str recurrent { ... } start time stop time [ ... ]
```

start | stop Defines the day, month, and year (*date*) in USA format (mm/dd/yyyy).
 Defines the hour and minutes (*time*) in the 24-hour clock format (hh:mm).

Examples: The following command creates a schedule definition named "mytime" which starts on 1/10/2003 at 11:00 AM and ends on 2/12/2003 at 7:00 PM:

```
set scheduler mytime once start 1/10/2003 11:00 stop 2/12/2003 19:00
```

The following command creates a schedule definition named "weekend" which starts at 8:00 AM and ends at 5:00 PM and repeats every Saturday and Sunday:

```
set scheduler weekend recurrent saturday start 8:00 stop 17:00  
set scheduler weekend recurrent sunday start 8:00 stop 17:00
```

scp

Description: Use the **scp** commands to configure the Secure Copy (SCP) client/server on NetScreen devices. SCP provides a way of transferring files to or from the NetScreen device using the SSH protocol.

Note: *It is possible to initiate file transfer from an external host, not from the NetScreen device itself.*

Syntax

get

`get scp`

set

`set scp enable`

unset

`unset scp enable`

Keywords and Variables

enable

`set scp enable`

`unset scp enable`

enable

Enables the Secure Copy (SCP) task. When SCP is enabled, the SSH task is activated if it is not already active.

service

Description: Use the **service** commands to create custom service definitions, modify existing service definitions, or display the current entries in the service definition list.

Use service definitions in policies to specify how the NetScreen device provides a service during a secure session. For example, a custom service definition might permit sessions using TCP protocol to exchange traffic between specified source and destination ports. Any policy that uses this definition conforms to these specifications.

Syntax

get

```
get service
[
  name_str |
  group [ name_str ] |
  pre-defined |
  timeout { other | tcp | udp } [ port number1 [ number2 ] ] |
  user
]
```

set

```
set service name_str
[
  +
  {
    icmp type number code number |
    ptcl_num | tcp | udp
    { src-port number-number dst-port number-number }
  } |
  protocol
  {
```

```
    ptcl_num | tcp | udp
      [ src-port number-number ]
        [ dst-port number-number [ timeout { number | never } ] ] |
    icmp type number code number |
  } |
  timeout { number | never }
]
```

unset

```
unset service name_str [ timeout ]
```

Keywords and Variables

Variable Parameters

```
get service name_str
set service name_str [ ... ]
unset service name_str
```

name_str Defines a name for the service.

+

```
set service name_str + { ... }
```

+

 Appends a service entry to the custom services list.

pre-defined

```
get service pre-defined
```

pre-defined Displays all the pre-defined services.

protocol

```
set service name_str protocol { ... } [ ... ]
```

- protocol** Defines the service by IP protocol.
Defines a protocol for the specified service.
- *ptcl_num* specifies the protocol by protocol number.
 - **icmp** specifies a ICMP-based service.
 - **type** identifies the ICMP message type, for example, “Destination Unreachable”.
 - **code** identifies a specific message from a ICMP message type group. For example, from the Destination Unreachable type group, there are various more specific messages identified by code such as Net Unreachable, Host Unreachable, Protocol Unreachable, and so on.
 - **tcp** specifies a TCP-based service.
 - **udp** specifies a UDP-based service.

Example: The following command sets a service named “ipsec” that uses protocol 50:

```
set service ipsec protocol 50
```

src-port | dst-port

```
set service name_str protocol { ... }  
[ src-port number-number ] [ dst-port number-number ]
```

- src-port** Defines a range of source port numbers valid for the service and protocol.
dst-port Defines a range of destination port numbers valid for the service and protocol.

Example: The following command sets a service named “test1” that uses destination tcp port 1001:

```
set service test1 protocol tcp src-port 0-65535 dst-port 1001-1001
```

timeout

```
get service timeout { other | tcp | udp } [ port number1 [ number2 ] ]
set service name_str timeout { number | never }
unset service name_str timeout
```

timeout Sets or displays the timeout value for session sessions created on a port for TCP, UDP, or other protocols. You can specify session timeout value (<number>) in minutes, or as **never**.

Example1: The following command sets a service named “telnet” with a timeout value of 10 minutes:

```
set service telnet timeout 10
```

Example2: The following command displays timeouts for UDP from port 1720 to 1800:

```
get service timeout udp port 1720 1800
```

user

```
get service user
```

user Displays all user-defined services.

Defaults

The default timeout for TCP connections is 30 minutes.

The default timeout for UDP connections is 1 minute.

Note: The maximum timeout value for TCP connections and UDP connections is 2160 minutes.

Using the **get service** command without any arguments displays all pre-defined, user-defined, and service group information in the service book.

session

Description: Use the **session** commands to clear or display entries in the session table of the NetScreen device.

The *session table* contains information about individual sessions between hosts that communicate through the NetScreen device. Because each session entry uniquely identifies two communicating hosts, it contains a unique combination of the following criteria:

- An individual IP address for the source host (no subnets with multiple addresses).
- An individual IP address for the destination host (no subnets with multiple addresses).
- An individual port number for the source host (not a range of ports).
- An individual port number for the destination host (not a range of ports).

Every time the NetScreen device initiates a new session, it creates a session entry and uses the information in the entry while processing subsequent traffic between the hosts.

The kind of session information listed by the **get session** command depends upon the NetScreen platform. (For example, on any NetScreen platform with a management module in slot 1, the **get session** command lists currently active sessions on that module.) Such sessions include management, log, and other administrative traffic. On any NetScreen device with one or more Secure Port Modules (SPMs), the **get session** command lists sessions that are active on the ASIC for each module. If a session crosses two ASICs, it counts as two sessions, one for each ASIC.

Syntax

clear

```
clear [ cluster ] session
[
  all |
  id id_num |
  [ src-ip ip_addr [ netmask mask ] ]
  [ dst-ip ip_addr [ netmask mask ] ]
  [ src-mac mac_addr ] [ dst-mac mac_addr ]
  [ protocol ptcl_num [ ptcl_num ] ]
]
```

```
        [ src-port port_num [ port_num ] ]
        [ dst-port port_num [ port_num ] ]
        [ vsd-id id_num ]
    ]

get

get session
    [
    id id_num |
    fragment |
    [ tunnel ]
        [ src-ip ip_addr [ netmask mask ] ]
        [ dst-ip ip_addr [ netmask mask ] ]
        [ src-mac mac_addr ] [ dst-mac mac_addr ]
        [ protocol ptcl_num [ ptcl_num ] ]
        [ src-port port_num [ port_num ] ]
        [ dst-port port_num [ port_num ] ]
    ]
```

Keywords and Variables

all

clear [**cluster**] **session all**

all Specifies all sessions.

cluster

clear cluster session [...]

cluster Propagates the **clear** operation to all other devices in a NSRP cluster.

id

```
clear [ cluster ] session id id_num
get session id id_num
```

id *id_num* Identifies a specific session with Session Identification number *id_num*.

Example: The following command displays the session table entry for the session with ID 5116:

```
get session id 5116
```

src-ip | dst-ip

```
clear [ cluster ] session [ src-ip ip_addr [ netmask mask ] ]
[ dst-ip ip_addr [ netmask mask ] ] [ ... ]
get session [ ... ] [ src-ip ip_addr [ netmask mask ] ]
[ dst-ip ip_addr [ netmask mask ] ] [ ... ]
```

src-ip *ip_addr* Identifies all sessions initiated by packets containing source IP address *ip_addr*. For example, *ip_addr* could be the source IP address in the first TCP SYN packet.

dst-ip *ip_addr* Identifies all sessions initiated by packets containing destination IP address *ip_addr*.

Example: The following command displays all the entries in the session table for a specific source IP address:

```
get session src-ip 172.16.10.92
```

src-mac | dst-mac

```
clear [ cluster ] session [ ... ] [ dst-ip ip_addr [ netmask mask ] ]
[ src-mac mac_addr ] [ dst-mac mac_addr ]
get session [ ... ] [ src-ip ip_addr [ netmask mask ] ]
[ dst-ip ip_addr [ netmask mask ] ]
```

src-mac Identifies all sessions initiated by packets containing source MAC address *mac_addr*.

dst-mac Identifies all sessions initiated by packets containing destination MAC address *mac_addr*.

protocol

```
clear [ cluster ] session [ ... ] protocol ptcl_num [ ptcl_num ] [ ... ]
get session [ ... ] protocol ptcl_num [ ptcl_num ] [ ... ]
```

protocol Identifies all sessions that use protocol *ptcl_num*.
You can also specify any protocol within a range (*ptcl_num ptcl_num*).

src-port | dst-port

```
clear [ cluster ] session [ ... ] [ src-port port_num [ port_num ] ]
[ dst-port port_num [ port_num ] ] [ ... ]
get session [ ... ] [ src-port port_num [ port_num ] ]
[ dst-port port_num [ port_num ] ]
```

src-port Identifies all sessions initiated by packets that contain the layer 4 source port *port_num* in the layer 4 protocol header.
You can also specify any layer 4 destination port within a range (*port_num port_num*).

dst-port Identifies all sessions initiated by packets that contain the layer 4 destination port *port_num* in the layer 4 protocol header.
You can also specify any layer 4 destination port within a range (*port_num port_num*).

Example: The following command displays all the entries in the session table for protocol 5 and for source ports 2 through 5:

```
get session protocol 5 src-port 2 5
```

tunnel

```
get session tunnel [ ... ]
```

tunnel Directs the NetScreen device to display tunnel sessions.

vsd-id

```
clear [ cluster ] session [ ... ] vsd-id id_num  
get session [ ... ] vsd-id id_num
```

vsd-id *id_num* Identifies all sessions that belong the VSD group *id_num*.

Example: The following command clears all sessions belonging to VSD group 2001, and initiated from the host at IP address 172.16.10.12:

```
clear session src-ip 172.16.10.12 vsd-id 2001
```

Description: Use the **sip** commands to configure SIP functionality on the NetScreen device and also to obtain information.

Syntax

get

```
get sip
{
  call |
  setting
}
```

set

```
set sip
{
  media-inactivity-timeout number |
  signaling-inactivity-timeout number
}
```

unset

```
unset sip
{
  media-inactivity-timeout |
  signaling-inactivity-timeout
}
```

Keywords and Variables

call

get sip call

call Displays the number of active calls. The maximum number of calls possible on a NetScreen device depends on the platform type. For more information, refer to the specifications sheet for your product.

media-inactivity-timeout

set sip media-inactivity-timeout *number*
unset sip media-inactivity-timeout

media-inactivity-timeout Configures or removes the maximum length of time (in seconds) a call can remain active without any media (RTP or RTCP) traffic within a group. Each time a RTP or RTCP packet occurs within a call, this timeout resets. The default setting is 120 seconds.

setting

get sip setting

setting Displays the inactivity timeout parameters of the SIP ALG (application-layer gateway) for SIP signaling and SIP media.

signaling-inactivity-timeout

set sip signaling-inactivity-timeout *number*
unset sip signaling-inactivity-timeout

signalling-inactivity-timeout Configures or removes the maximum length of time (in seconds) a call can remain active without any SIP signaling traffic. Each time a SIP signaling message occurs within a call, this timeout resets. The default setting is 43200 seconds (12 hours).

snmp

Description: Use the **snmp** commands to configure the NetScreen device for Simple Network Management Protocol (SNMP), to gather statistical information from the NetScreen device, and receive notification when significant events occur.

Syntax

get

```
get snmp [ auth-trap | community name_str | settings ]
```

set

```
set snmp
{
  auth-trap enable |
  community name_str
    { read-only | read-write }
    [
      trap-off |
      trap-on [ traffic ] |
      version { any | v1 | v2c }
    ] |
  contact name_str |
  host comm_name ip_addr[/mask ]
    [
      src-interface interface |
      trap { v1 | v2c }
    ] |
  location string |
  name name_str |
  port { listen [ port_num ] | trap [ port_num ] } |
}
```

unset

```
unset snmp
{
  auth-trap enable |
  community name_str |
  contact |
  host comm_name ip_addr [ src-interface ] |
  location |
  name |
  port { listen [ port_num ] | trap [ port_num ] } |
}
```

Keywords and Variables

auth-trap enable

```
get snmp auth-trap
set snmp auth-trap enable
unset snmp auth-trap enable
```

auth-trap enable Enables Simple Network Management Protocol (SNMP) authentication traps.

community

```
get snmp community name_str
set snmp community name_str { ... }
unset snmp community name_str
```

community Defines the name for the SNMP community. It supports maximum 3 communities in all products.

- **read-only** Defines the permission for the community as “read-only.”
- **read-write** Defines the permission for the community as “read-write.”

- **trap-off** Disables SNMP traps for the community.
- **trap-on** Enables SNMP traps for the community. The **traffic** switch includes traffic alarms as SNMP traps.

Examples: The following command configures a community named “public”.

- Allows hosts to read MIB data from the SNMP agent
- Enables SNMP traps for the community

```
set snmp community public read-only trap-on
```

The following command configures an SNMP host with IP address 10.20.25.30 for the community named “public”:

```
set snmp host public 10.20.25.30
```

contact

```
set snmp contact name_str  
unset snmp contact
```

contact Defines the system contact.

host

```
set snmp host comm_name ip_addr[/mask ] [ ... ]  
unset snmp host comm_name ip_addr [ ... ]
```

host Defines the community name string and the IP address of the SNMP management host. The *mask* value defines a SNMP community member as a subnet. Note: When you define an SNMP community member as a subnet, that member can poll the NetScreen device but it cannot receive SNMP traps. To receive SNMP traps, the community member must be a single host.

Example: The following commands configure a community named “netscreen.”

- Specifies read and write permission
- Allows the NetScreen device to send traps to all hosts in the community
- Assigns the community to an SNMP host with IP address 10.40.40.15

```
set snmp community netscreen read-write trap-on
set snmp host netscreen 10.40.40.15
```

Example: The following command defines the subnet 10.5.1.0/24 as a member of the SNMP community named olympia:

```
set snmp host olympia 10.5.1.0/24
```

location

```
set snmp location string
unset snmp location
```

location Defines the physical location of the system.

name

```
set snmp name name_str
unset snmp name
```

name Defines the name of the system.

port

```
set snmp port { ... }
unset snmp port { ... }
```

port Specifies the SNMP listen and trap port (**listen** | **trap**).

settings

get snmp settings

settings Displays the name of the contact person, and the name and physical location of the NetScreen device.

src-interface

```
set snmp host comm_name ip_addr[/mask] src-interface interface  
unset snmp host comm_name ip_addr[/mask] src-interface
```

src-interface Specifies the source interface.

trap

```
set snmp host comm_name ip_addr[/mask] trap v1 | v2c
```

trap If an SNMP community supports both SNMP versions (SNMPv1 (**v1**) and SNMPv2c (**v2c**), you must specify a trap version for each community member.

version

```
set snmp community { ... } version { any | v1 | v2c }
```

version When you create an SNMP community, you can specify whether the community supports SNMPv1 (**v1**), SNMPv2c (**v2c**), or both SNMP versions, as required by the SNMP management stations. For backward compatibility with earlier ScreenOS releases that only support SNMPv1, NetScreen devices support SNMPv1 by default.

socket

Description: Use the **socket** commands to display socket information on a NetScreen device.

A *socket* is a software object that serves as a connection to a network protocol. A NetScreen device can send and receive TCP/IP or UDP traffic by opening a socket and reading and writing data to and from the socket.

Syntax

get

```
get socket [ id id_num ]
```

Keywords and Variables

id

```
get socket id id_num
```

id Displays the information for an identified socket (*id_num*).

Example: The following command displays the information concerning socket 5:

```
get socket id 5
```

ssh

Description: Use the **ssh** commands to configure the Secure Shell (SSH) server task.

The SSH server task is an SSH-compatible server application that resides on the NetScreen device. When you enable the SSH server task, SSH client applications can manage the device through a secure connection. (The look and feel of a SSH client session is identical to a Telnet session.) You can run either SSH version 1 (SSHv1) or SSH version 2 (SSHv2) on the NetScreen device; the commands available depend on the SSH version that you activate.

Syntax

clear

```
clear ssh
{
  all |
  enables |
  host-key |
  pka-key |
  sessions
}
```

exec (SSHv1)

```
exec ssh tftp pka-rsa [ user-name name_str ] file-name filename ip-addr ip_addr
[ from interface ]
```

exec (SSHv2)

```
exec ssh tftp pka-dsa [ user-name name_str ] file-name filename ip-addr ip_addr
[ from interface ]
```

get (SSHv1)

```
get ssh
[
  host-key |
  pka-rsa [ all | [ username name_str ] [ index number ] ] |
  report
]
```

get (SSHv2)

```
get ssh
[
  host-key |
  pka-dsa [ all | [ user-name name_str ] [ index number ] ] |
  report
]
```

set (SSHv1)

```
set ssh
{
  enable |
  key-gen-time number |
  pka-rsa [ username name_str ] key number1 number2 number3
}
```

set (SSHv2)

```
set ssh
{
  enable |
  pka-dsa
  {
    user-name name_str { key string | pka-key-id string } |

```

```
    key string
  } |
  pub-key string |
  version { v1 | v2 }
}
```

unset (SSHv1)

```
unset ssh
{
  enable |
  host-key |
  key-gen-time |
  pka-rsa
  {
    all |
    username name_str
    {
      all |
      index id_num
    }
  }
}
```

unset (SSHv2)

```
unset ssh
{
  enable |
  host-key |
  pka-dsa
  {
    all |
    user-name name_str
  }
}
```

```
        all |
        index id_num
      }
    } |
    pub-key string
  }
```

Keywords and Variables

all

```
clear ssh all
```

all Clear all SSH sessions, enables, PKA keys, and host keys on device.

enable

```
set ssh enable
unset ssh enable
```

enable Enables the Secure Shell (SSH) task. When issued from a vsys, enables SSH for the vsys.

host-key

```
get ssh host-key
unset ssh host-key
```

host-key The **get** command shows the SSH host key (RSA public key for SSHv1 and DSA public key for SSHv2) for the root or current vsys, including the fingerprint of the host key. The **clear** command deletes the SSH host key for the root or current vsys; SSH must be disabled first before you can delete the host key.

key-gen-time

```
set ssh key-gen-time number  
unset ssh key-gen-time
```

key-gen-time Specifies the SSHv1 server key regenerating time (in minutes).

pka-dsa

```
get ssh pka-dsa [ ... ]  
set ssh pka-dsa [ ... ]  
unset ssh pka-dsa { ... }
```

pka-dsa Public Key Authentication (PKA) using Digital Signature Algorithm (DSA) for SSHv2.

- **all** Shows all PKA public keys bound to all users. You must be the root user to execute this option; read-write users and read-only users cannot execute this command.
- **index** *number* allows the admin user and read-only user to view the details of a key bound to the active admin. It also allows the root user to view the details of a key bound to the specified user.
- **key** *string* Binds a PKA key to the current user. Read-only users cannot execute this option.
- **pka-key-id** *string* Binds a PKA key identified by the key ID to the current user. Read-only users cannot execute this option.
- **user-name** *name_str* Specifies the name of the user to bind the PKA key. **file-name** *filename* Specifies the file containing the key to bind to the user. For the get command, **user-name** displays all PKA public keys bound to a specified user *name_str*. Admin users and read-only users can execute this option only if *name_str* identifies the current admin user or read-only user.

Example: The following command binds a hypothetical key to a user named “chris”:

```
set ssh pka-dsa user-name chris key AAAAB3NzaC1kc3MAAABBAPrdVkvpsilMT7NfZJm24pqMU2
  FFpO49+LFmbOiPljEYelWTA4J5...
```

The following command:

- loads a key contained in a file named “key_file”
- takes the file from a server at IP address 172.16.10.11
- binds the key to a user named “chris”

```
exec ssh tftp pka-dsa user-name chris file-name key_file ip-addr 172.16.10.11
```

pka-key

```
clear ssh pka-key
```

pka-key Deletes all SSH PKA keys on the device.

pka-rsa

```
get ssh pka-rsa [ ... ]
set ssh pka-rsa [ ... ]
unset ssh pka-rsa { ... }
```

pka-rsa Public Key Authentication (PKA) using RSA for SSHv1.

- **all** Shows all PKA public keys bound to all users. You must be the root user to execute this option; admin users and read-only users cannot execute this command.
- **index number** allows the admin user and read-only user to view the details of a key bound to the active admin. It also allows the root user to view the details of a key bound to the specified user.

- **key** *number1 number2 number3* Binds a PKA key to the current user. The *number1*, *number2*, and *number3* values represent the key length, the exponent, and the modulus, respectively. Read-only users cannot execute this option.
- **username** *name_str* Specifies the name of the user to bind the PKA key. **file-name** *filename* Specifies the file containing the key to bind to the user. For the get command, **username** displays all PKA public keys bound to a specified user *name_str*. Admin users and read-only users can execute this option only if *name_str* identifies the current admin user or read-only user.

Example: The following command binds a hypothetical key to a user named “chris”:

```
set ssh pka-rsa username chris key 512 655376875272488448958071956054093391935
033213724615582796813757422715643970626128793365599992658289801116115376527150
77837089019119296718115311887359071551679
```

The following command loads a key.

- Key contained in a file named “key_file”
- File taken from a server at IP address 172.16.10.11
- Key bound to a user named “chris”

```
exec ssh tftp pka-rsa username chris file-name key_file ip-addr 172.16.10.11
```

pub-key

```
set ssh pub-key string
unset ssh pub-key string
```

pub-key	Sets the public key for SSHv2.
----------------	--------------------------------

report

```
get ssh report
```

report	Displays SSHv1 (or SSHv2) key, session, and vsys information for the device on which SSH is currently enabled.
---------------	--

sessions

clear ssh sessions

sessions Logs out all admins that currently have active SSH sessions.

version

set ssh version v1 | v2

version (Available only at the root level.) Sets the version of SSH on the NetScreen device. Specify either SSH version 1 or version 2. Before you can set an SSH version, make sure that all keys created with the previous version are removed by executing the **delete ssh device all** command. To clear SSHv2 keys; issue the **clear scs all** command to clear SSHv1 keys.

Defaults

This feature is disabled by default.

The default key generation time for SSHv1 is 60 minutes.

Description: Use the **ssl** commands to configure a Secure Sockets Layer (SSL) connection, or to display the SSL configuration on a NetScreen device.

Secure Sockets Layer (SSL) is a set of protocols that can provide a secure connection between a Web client and a Web server communicating over a TCP/IP network.

Syntax

get

```
get ssl [ ca-list | cert-list ]
```

set

```
set ssl
{
  cert number |
  enable |
  encrypt { { 3des | des } sha-1 | { rc4 | rc4-40 } md5 }
  port port_num
}
```

unset

```
unset ssl { cert | enable | encrypt | port }
```

Keywords and Variables

ca-list | cert-list

```
get ssl ca-list  
get ssl cert-list
```

ca-list | cert-list Displays currently configured Certificate Authorities (**ca-list**) or currently available certificates (**cert-list**).

Example: The following command displays the SSL certificate list:

```
get ssl cert-list
```

cert

```
set ssl cert number  
unset ssl cert
```

cert Specifies that the named certificate is required.

enable

```
set ssl enable  
set ssl enable  
unset ssl enable
```

enable Turns on SSL.

encrypt

```
set ssl encrypt
  { 3des | des } sha-1 |
  { rc4 | rc4-40 } md5
unset ssl encrypt
```

encrypt Enables encryption over the SSL connection.

- **3des** Set the 3DES security level.
- **des** Sets the DES security level.
- **rc4 md5** Sets the RC4 MD3 security level.
- **rc4-40 md5** Sets the RC4-40 MD3 security level.

Example: The following command specifies triple-DES encryption with SHA-1 authentication hashing:

```
set ssl encrypt 3des sha-1
```

port

```
set ssl port port_num
unset ssl port
```

port Specifies the SSL port number.

Example: The following command changes the SSL port to 11533:

```
set ssl port 11533
```

Defaults

The default SSL port is 443.

sys-clock

Description: Use the **sys-clock** command to display information on the system clock.

Syntax

get

`get sys_clock`

Keywords and Variables

None.

syslog

Description: Use the **syslog** commands to configure the NetScreen device to send traffic and event messages to up to four syslog hosts, or to display the current syslog configuration.

Note: *The syslog host must be enabled before you can enable syslog.*

Syntax

get

```
get syslog
```

set

```
set syslog
{
  config { name_str | ip_addr }
  [
    facilities
      AUTH/SEC |
      local0 | local1 | local2 | local3 | local4 | local5 | local6 | local7
      {
        AUTH/SEC |
        local0 | local1 | local2 | local3 | local4 | local5 | local6 | local7
      } |
    log { all | event | traffic } |
    port number /
    transport tcp
  ] |
  enable |
  src-interface interface |
}
```

unset

```
unset syslog
  config [ name_str | ip_addr ]
    {
      log all | event | traffic |
      port
      transport
    }
  enable |
  src-interface
```

Keywords and Variables

config

```
set syslog config { name_str | ip_addr } { ... }
unset syslog config [ ip_addr ]
```

config Defines the configuration settings for the syslog utility. The { *name_str* | *ip_addr* } parameters define the name or the IP address of the syslog host device. You can define up to four syslog hosts. Specifying an IP address with the unset syslog config command removes the configuration for the specified syslog host. Otherwise, this command removes the configuration for all syslog hosts.

enable

```
set syslog enable
unset syslog enable
```

enable Enables the NetScreen device to send messages to the syslog host(s).

facility

```
set syslog config { name_str | ip_addr } facility
{
  AUTH/SEC |
  local0 | local1 | local2 | local3 | local4 | local5 | local6 | local7
}
{
  AUTH/SEC |
  local0 | local1 | local2 | local3 | local4 | local5 | local6 | local7
}
```

facility Defines the *security facility level* and the *regular facility level* for each syslog host that you specify. The security facility classifies and sends messages to the syslog host for security-related actions such as attacks. The regular facility classifies and sends messages for events unrelated to security, such as user logins and logouts, and system status reports.

Example: The following command sets the syslog host configuration to report all logs:

```
set syslog config 172.16.20.249 facility local0 local1
```

log

```
set syslog config { name_str | ip_addr } log all | event | traffic
unset syslog config { name_str | ip_addr } log all | event | traffic
```

log Directs the NetScreen device to send traffic log entries, event log entries or both traffic and event log entries to the syslog host.

port

```
set syslog config { name_str | ip_addr } port port_num  
unset syslog config { name_str | ip_addr } port
```

port Defines the port number on the syslog host that receives the User Datagram Protocol (UDP) packets from the NetScreen device.

Example: The following command changes the syslog port number to 911:

```
set syslog config port 911
```

src-interface

```
set syslog config { name_str | ip_addr } src-interface interface  
unset syslog config { name_str | ip_addr } src-interface
```

src-interface Specifies the source interface.

transport tcp

```
set syslog config { ip_addr | name_str } transport tcp
```

transport tcp Directs the device to use TCP protocol instead of UDP protocol.

Defaults

This feature is disabled by default. The default syslog port number is 514, and the default WebTrends port number is 514.

system

Description: Use the **get system** command to display general system information.

The information displayed by the **get system** command includes:

- Descriptive indices of the ScreenOS operating system, including Serial Number, Control Number, Software Number, and image source file name.
- Descriptive indices of the NetScreen hardware platform, including hardware version, MAC address, and type.
- Chronological and timekeeping information.
- Current operational mode (transparent, NAT, or route)
- Configuration port and user IP.
- Interface settings.

Syntax

get

get system

Keywords and Variables

None.

tech-support

Description: Use the **tech-support** command to display system information.

The information displayed by the **get tech-support** command is useful for troubleshooting the NetScreen device. Most of this information consists of the current authentication and routing settings.

Syntax

get

get tech-support

Keywords and Variables

None.

temperature-threshold

Description: Use the **temperature-threshold** commands to set the normal and severe temperature thresholds for triggering temperature alarms.

Syntax

set

```
set temperature-threshold
    { alarm | severe }
    { celsius number | fahrenheit number }
```

unset

```
unset temperature-threshold
    { alarm | severe }
    { celsius number | fahrenheit number }
```

Arguments

alarm | severe Defines the temperature (**celsius** or **fahrenheit**) required to trigger a regular alarm or a severe alarm. A severe alarm sounds a greater frequency of audible alarms and a generates greater number of event log entries.

Example: To enable the audible alarm to sound when the temperature of the device reaches or exceeds 190 degrees fahrenheit:

```
set temperature-threshold alarm fahrenheit 190
```

timer

Description: Use the **timer** commands to display timer settings, or to configure the NetScreen device to automatically execute management or diagnosis at a specified time.

All timer settings remain in the configuration script after the specified time has expired.

Syntax

get

```
get timer
```

set

```
set timer date time action reset
```

unset

```
unset timer id_num
```

Keywords and Variables

Variable Parameters

```
set timer date time action reset
```

```
unset timer id_num
```

<i>date</i>	Specifies the date when the NetScreen device executes the defined action. Date is in <i>mm/dd/yyyy</i> format.
<i>time</i>	Specifies the time when the NetScreen device executes the defined action. Time is in <i>hh:mm</i> format.
<i>id_num</i>	Identifies a specific action by ID number in the list of timer settings (generated by the set timer command.)

action

```
set timer date time action reset
```

action Defines the event that the command triggers at the given date and time.

reset

```
set timer date time action reset
```

reset Resets the timer.

Example: The following command configures NetScreen to reset at a given time and date:

```
set timer 1/31/2000 19:00 action reset
```

trace-route

Description: Use the **trace-route** command to display the route to a host.

Syntax

```
trace-route { ip_addr | name_str } [ hop number [ time-out number ] ]
```

Keywords

Variable Parameters

```
trace-route ip_addr  
trace-route name_str
```

ip_addr | *name_str* The IP address (*ip_addr*) or object name (*name_str*) of the host.

hop

```
trace-route { ip_addr | name_str } hop number [ ... ]
```

hop The maximum number of trace route hops (*number*) to evaluate and display.

Example: The following command:

- evaluates and displays up to four route trace hops
- sends the output to a host with IP address 172.16.10.10

```
trace-route 172.16.10.10 hop 4
```

time-out

trace-route { *ip_addr* | *name_str* } **hop** *number* **time-out** *number*

time-out Specifies the amount of time in seconds (*number*) to elapse before abandoning the route trace.

Example: The following command performs a trace-route operation.

- Evaluates and displays up to four route trace hops
- Sends the output to a host with IP address 172.16.10.10
- Specifies a timeout value of four seconds

trace-route 172.16.10.10 hop 4 time-out 4

traffic-shaping

Description: Use the **traffic-shaping** commands to determine the settings for the system with the traffic-shaping function, or to display information on traffic management device interfaces.

Traffic shaping is the allocation of the appropriate amount of network bandwidth to every user and application on an interface. The appropriate amount of bandwidth is defined as cost-effective carrying capacity at a guaranteed Quality of Service (QoS). You can use a NetScreen device to shape traffic by creating policies and by applying appropriate rate controls to each class of traffic going through the NetScreen device.

Note: You can only apply traffic shaping to policies for which the destination zone has a single bound interface.

Syntax

get

```
get traffic-shaping
{
  interface [ interface ] |
  ip_precedence |
  mode
}
```

set

```
set traffic-shaping
{
  ip_precedence
    number1 number2 number3 number4 number5 number6 number7 number8 |
  mode { auto | off | on }
}
```

unset

```
unset traffic-shaping { ip_precedence | mode }
```

Keywords and Variables

interface

```
get traffic-shaping interface [ interface ]
```

interface Displays the traffic shaping info for an interface.

ip_precedence

```
get traffic-shaping ip_precedence
set traffic-shaping ip_precedence
    number1 number2 number3 number4 number5 number6 number7 number8
unset traffic-shaping mode ip_precedence
```

ip_precedence Specifies the Priorities 0 through 7 for IP precedence (TOS) mapping. Each setting should be a single-digit value.

mode

```
get traffic-shaping mode
set traffic-shaping mode { auto | off | on }
unset traffic-shaping mode
```

mode Defines the mode settings for the system with the traffic-shaping function. If you select **auto**, the system automatically determines the mode settings. If there is at least one policy in the system with traffic-shaping turned on, the system automatically sets the mode to **on**. If there is no such policy, the auto mode default setting is **off**.

Defaults

By default, the traffic shaping function is set up to automatic mode.

url

Description: Use the **url** commands to enable or disable URL filtering for use in policies, and to configure and display URL filter settings.

Note: A Websense server provides the URL filtering.

Syntax

get

```
get url [ all | vsys-name vsys_name ]
```

set (root level)

```
set url
{
  config { disable | enable } |
  fail-mode { block | permit } |
  message string |
  server { ip_addr | dom_name } port_num number |
  src-interface interface |
  type { NetScreen | Websense }
}
```

set (vsys level)

```
set url
{
  config { disable | enable } |
  use-root |
  use-vsys
}
```

unset

```
unset url
{
  config |
  fail-mode |
  message |
  server |
  src-interface |
  type
}
```

Keywords and Variables

config

```
set url config { disable | enable }
unset url config
```

config { disable enable }	Disables or enables URL filtering at the device level for use in policies. By itself, enabling URL filtering at the device level does not activate it. You must enable URL filtering at the device and policy level to apply filtering to URL requests.
---	---

fail-mode

```
set url fail-mode { block | permit }
unset url fail-mode
```

fail-mode { block permit }	If the connection between the NetScreen device and the Websense server is lost, the NetScreen device either blocks or permits all HTTP requests to which a policy requiring URL filtering applies. The default fail-mode behavior is to block HTTP requests.
--	--

message

```
set url message string  
unset url message
```

message *string* Defines a custom message, 1-500 characters in length, to send to the client who is blocked from reaching a URL.

Example: The following command defines the URL blocking message “This site is blocked”:

```
set url message "This site is blocked."
```

server

```
set url server { ip_addr | dom_name } port_num number  
unset url server
```

server Defines the following connection parameters for the URL filtering server:

- *ip_addr* | *dom_name* Sets the IP address or DNS name of the URL filtering server.
- *port_num* Sets the port number on which the NetScreen device communicates with the URL filtering server. The default port number is 15868.
- *number* Sets the timeout interval, in seconds, that the NetScreen device waits for a response from the Websense filter. If Websense does not respond within the time interval, the NetScreen device either blocks the request or allows it, as you choose. The default is 10 seconds.

Example: The following command sets the IP address, port number, and timeout value for the URL filtering server (the port number and timeout interval use the default values):

```
set url server 1.2.2.20 15868 10
```

src-interface

```
set url src-interface interface  
unset url src-interface
```

src-interface <i>interface</i>	Specifies the source interface that the NetScreen device uses when communicating with the Websense server. If you specify a source interface, the device enforces use of that interface, without consulting the routing table. If you do not specify an interface, the device picks an interface according to entries in the routing table.
--	---

type

```
set url type { NetScreen | Websense }  
unset url type
```

type { NetScreen Websense }	Specifies the source of the message that the NetScreen device delivers to clients when URLs are blocked—the NetScreen device or the Websense server.
--------------------------------------	--

use-root

```
set url use-root
```

use-root	When entering this command in a virtual system (vsys), it instructs the vsys to share the URL filtering server defined at the root level.
-----------------	---

use-vsys

```
set url use-vsys
```

use-vsys	When entering this command in a virtual system (vsys), it instructs the vsys to use the URL filtering server defined for that vsys.
-----------------	---

user

Description: Use the **user** commands to create, remove, or display entries in the internal user authentication database.

The basic user categories are as follows:

- Authentication users (for using network connections)
- IKE users (for using AutoKey IKE VPNs)
- L2TP users (for using L2TP tunnels)
- XAuth users

Syntax

get

```
get user { name_str | all | id id_num }
```

set

```
set user name_str
{
  disable |
  enable |
  hash-password string |
  ike-id
  {
    asn1-dn { [ container string ] wildcard string } [ share-limit number ] |
    fqdn name_str |
    ip ip_addr |
    u-fqdn name_str
  } |
  password pswd_str |
```

```
remote-settings
{
  dns1 ip_addr |
  dns2 ip_addr |
  ipaddr ip_addr |
  ippool name_str |
  wins1 ip_addr |
  wins2 ip_addr
} |
type { [ auth ] [ ike ] [ l2tp ] [ xauth ] } |
uid id_num
}
```

unset

```
unset user name_str | id_str
[
  remote-settings { dns1 | dns2 | ipaddr | ippool | wins1 | wins2 } |
  type [ auth ] [ ike ] [ l2tp ] [ xauth ]
]
```

Keywords and Variables

Variable Parameters

```
get user name_str
set user name_str { ... }
unset user name_str [ ... ]
```

user Defines the user's name.

Examples: The following command displays information for a user named “roger”:

```
get user roger
```


The following command deletes the user named **jane**:

```
unset user jane
```

all

```
get user all
```

all Displays the following information for all the entries in the internal user database:

- User ID number
- User name
- Status (enabled or disabled)
- User type
- IKE ID types – e-mail address, IP address, or domain name—and IKE identifies

disable | enable

```
set user name_str disable
```

```
set user name_str enable
```

disable | enable Disables or enables the user in the internal database. By default, the user is disabled. If you set a password for an auth user or an IKE ID for an IKE user, the user becomes enabled automatically.

id

```
get user id id_num
```

id Displays information on the user, identified by `id_num`. This option displays the same information as **get user name_str** option.

Example: The following command displays a particular user with user ID “10”:

```
get user id 10
```

hash-password

```
set user name_str hash-password string
```

hash-password Creates a hashed password for the specified user and stores it in the configuration. Only an auth user can have a hashed password. The NetScreen device generates a hashed password randomly using either the crypt () or SHA-1 algorithm.

ike-id

```
set user name_str ike-id { ... }
```

ike-id { *ip_addr* | *name_str* } Adds and defines an AutoKey IKE dialup user.

- **ip** *ip_addr* The IP address of the dialup user.
- **fqdn** *name_str* The Fully Qualified Domain Name, the complete string, such as www.juniper.net.
- **u-fqdn** *name_str* Specifies the dialup user identity, usually equivalent to an e-mail address such as admin@acme.com.
- **asn1-dn** Specifies the user certificate distinguished name fields, and field values that define user identity.
 - **container** *string* Specifies a container identity. This identity allows multiple identity fields for each type (CN, OU, O, L, ST, C, and E). To match a local ASN1_DN identity, the peer IKE identity fields must match all identity fields specified in the container identity. The NetScreen device does not check any undefined container fields. Field sequence must be identical.
 - **wildcard** *string* Specifies a wildcard identity. This identity allows only one identity field for each type (CN, OU, O, L, ST, C, and E). To match a local ASN1_DN identity configuration, the peer IKE identity must contain fields matching all non-empty identity fields specified in the wildcard identity. For example, the wildcard identity **o=ACME,ou=Marketing** allows tunnel communication with any user whose certificate contains these field values. The NetScreen device does not check any undefined wildcard fields. Field sequence is not important.
- **share-limit** *number* Specifies the number of users that can establish tunnels concurrently using this identity. When this number is larger than 1, the NetScreen device treats it as a Group IKE ID user. With Group IKE ID, multiple dialup users can establish tunnels using partial IKE identities.

Examples: The following command creates an IKE user named **branchsf** with the IKE-ID number 2.2.2.2:

```
set user branchsf ike-id ip 2.2.2.2
```

The following command creates a new user definition named “market”.

- Configures the user definition to recognize up to 10 hosts
- Specifies that the hosts must possess certificates containing “ACME” in the O field, and “Marketing” in the OU field

```
set user market ike-id asn1-dn wildcard "o=ACME,ou=Marketing" share-limit 10
```

(This command uses Group IKE ID, which allows multiple hosts to use a single user definition. For more information on Group IKE ID, see the *NetScreen Concepts & Examples ScreenOS Reference Guide*.)

password

```
set user name_str password pswd_str
```

password Defines a top-level password, used to authenticate the auth, L2TP, IKE or XAuth user.

Example: The following command creates an authentication user in the NetScreen internal database for user **guest** with the password **JnPc3g12**:

```
set user guest password JnPc3g12
```

remote-settings

```
set user name_str remote-settings { dns1 ip_addr | dns2 ip_addr }
set user name_str remote-settings ipaddr ip_addr
set user name_str remote-settings ippool name_str |
set user name_str remote-settings { wins1 ip_addr | wins2 ip_addr }
```

remote-settings Sets the remote settings for the user.

- **dns1 | dns2** Specifies the IP address of the primary and secondary DNS servers.
- **ipaddr** Specifies the static IP address for the user.

- **ippool** Specifies the named L2TP IP pool, which contains a range of IP addresses. The NetScreen device uses IP pools when it assigns addresses to dialup users using L2TP. (To define a L2TP pool, use the **set ippool** command.)
- **wins1 | wins2** Specifies primary and secondary servers that provide WINS (Windows Internet Naming Service). WINS is a service for mapping IP addresses to NetBIOS computer names on Windows NT server-based networks. A WINS server maps a NetBIOS name used in a Windows network environment to an IP address used on an IP-based network.

Example: The following command directs the device to obtain an IP address from an L2TP ippool named NY_Pool for a dialup user named John_Doe:

```
set user John_Doe remote-settings ippool NY_Pool
```

type

```
set user name_str type { [ auth ] [ ike ] [ l2tp ] [ xauth ] }
```

type Sets the user type, in any of the following combinations:
auth, ike, l2tp, xauth, auth ike l2tp xauth, auth ike, auth l2tp, auth xauth, ike l2tp, ike xauth, l2tp xauth, auth ike l2tp, auth l2tp xauth, or ike l2tp xauth.

Example: The following command changes the user **guest** to an authentication/L2TP user:

```
set user guest type auth l2tp
```

user-group

Description: Use the **user-group** commands to create or delete a user group, to modify it, or to add or remove a user from it.

User groups allow policies to treat multiple users in the same way, thus avoiding individual configurations for individual users. For example, even though you can configure dialup VPN tunnels for IKE users on a per-user basis, it is often more efficient to aggregate the users into a group, for which only one tunnel configuration is necessary.

Any policy that references a user group applies to all the members in the group. An authentication user can be a member of up to four different user groups.

Note: Different NetScreen platforms allow a different number of members in a user group.

Syntax

get

```
get user-group { name_str | all | external | id id_num | local }
```

set

```
set user-group name_str
{
  id id_num |
  location { external | local } |
  type { [ auth ] [ ike ] [ l2tp ] [ xauth ] } |
  user name_str
}
```

unset

```
unset user-group
{
  name_str [ location | type | user name_str ] |
  id id_num
}
```

Keywords and Variables

Variable Parameter

```
get user-group name_str
set user-group name_str { ... }
unset user-group name_str [ ... ]
```

name_str Specifies the name of the user group.

Example: The following command displays the contents of a user group named Corp_Dial:

```
get user-group Corp_Dial
```

all

```
get user-group all
```

all Displays all existing user groups.

external

```
get user-group external
set user-group name_str location external
```

external Defines a user group as external. You can store user definitions in groups on an external RADIUS server. You can then define a user group on the NetScreen device, define the type of user it contains, leave it unpopulated of users, and define the user group as external. Defining an external user group on the NetScreen device allows you to reference that group in policies requiring authentication. When the policy requires an authentication check, the NetScreen device then contacts the RADIUS server, which performs the authentication check.

id

```
get user-group id id_num
set user-group name_str id id_num
unset user-group name_str [ ... ]
```

id Identifies the user group with an identification number *id_num*.

```
unset user-group name_str [ ... ]
```

name_str Specifies the name of the user group.

Example: The following command creates a user group named Corp_Dial, and assigns the group an ID of 10:

```
set user-group Corp_Dial id 10
```

local

```
get user-group local
```

local Displays all local user groups.

location

```
set user-group name_str location { external | local }  
unset user-group name_str location
```

- location** Specifies the location of the user group:
- **external** Indicates that the user group is stored on an external authentication server. (NetScreen supports user groups on RADIUS servers.)
 - **local** Indicates that the user group is stored in the local database on the NetScreen device.

type

```
set user-group name_str type { ... }
```

- type** Specifies the type of user group when that group is stored on an external RADIUS server. (When the user-group is stored in the local database, the user types determine the type of user group.) The following are the possible user group types:
- **auth** Specifies that the group is comprised of authentication users.
 - **ike** Specifies that the group uses IKE.
 - **l2tp** Specifies L2TP users.
 - **xauth** Specifies XAuth users.

user

```
set user-group name_str user name_str  
unset user-group name_str user name_str
```

- user *name_str*** Adds or removes the named user to the specified user group.

Examples: The following commands create a new authentication user named “guest”.

- A authentication user group named “Corp_Dial” with ID 1010
- User added to the user group

```
set user guest password JnPc3g12
set user-group Corp_Dial location local
set user-group Corp_Dial user guest
```

The following command removes the user **guest** from the group:

```
unset user-group Corp_Dial user guest
```

Description: Use the **vip** commands to display the virtual IP (VIP) address configuration settings and to enable all VIPs to support multi-port services.

A *virtual IP* (VIP) address maps traffic received at one IP address to another address based on the destination port number in the TCP or UDP segment header.

Syntax

get

```
get vip
[
  ip_addr { port port_num | port-status } |
  server |
  session timeout
]
```

set

```
set vip
{
  multi-port |
  session timeout number
}
```

Keywords and Variables

Variable Parameters

```
get vip ip_addr { port port_num | port-status }
```

ip_addr Identifies the VIP address.

port *port_num* Identifies the destination port, so that the NetScreen device can display information about the specified virtual port defined on the VIP.

port-status Displays information about port allocation on the specified VIP.

multi-port

```
set vip multi-port
```

multi-port Enables the support of multiple virtual ports per custom service. By default, VIPs support single-port services. **Warning:** After you execute this command, you must restart the device. This command changes the functionality of the VIP, so it is highly inadvisable to switch back and forth between enabling and disabling the **multi-port** modes.

server

```
get vip server
```

server Displays the connectivity status of servers receiving traffic via VIPs.

session

```
get vip session
```

session timeout Displays the outstanding session timeout value for VIP.

Description: Use the **vpn** commands to create or remove a Virtual Private Network (VPN) tunnel, or to display current VPN tunnel parameters.

A *tunnel* is a way to secure VPN communication across a WAN. The tunnel consists of a pair of unidirectional security associations (SAs), one at each end of the tunnel, that specify the security parameter index (SPI), destination IP address, and security protocol (Authentication Header or Encapsulating Security Payload) used to exchange packets through the tunnel.

Juniper Networks NetScreen devices support two keying methods for establishing VPN tunnels, AutoKey IKE and Manual Key. AutoKey IKE (Internet Key Exchange) is a standard protocol that automatically establishes and maintains encryption keys between the participants. Manual Key VPNs use predefined keys that remain unchanged until the participants change them explicitly.

Syntax

get

```
get vpn [ name_str [ detail ] | auto | manual | proxy-id | ]
```

set (AutoKey IKE)

```
set vpn tunn_str gateway { ip_addr | name_str }
    [ replay | no-replay ]
    [ transport | tunnel ]
    [ idletime number ]
    {
        proposal [ name_str1 [ name_str2 [ name_str3 [ name_str4 ] ] ] ] |
        sec-level { basic | compatible | standard }
    }
```

set (Manual Key)

```
set vpn tunn_str manual spi_num1 spi_num2 gateway ip_addr
[ outgoing-interface interface ]
{
  ah { md5 | sha-1 }
    { key key_str | password pswd_str } |
  esp
    {
      aes128 | aes192 | aes256 | des | 3des
        { key key_str | password pswd_str } |
      null
    }
    [ auth { md5 | sha-1 }
      { key key_str | password pswd_str }
    ]
}
```

Other set Commands

```
set vpn tunn_str
{
  bind { interface interface | zone name_str } |
  df-bit { clear | copy | set } |
  failover-weight number
  monitor [ source-interface interface [ destination-ip ip_addr ] ]
    [ optimized ] [ rekey ] |
  proxy-id local-ip ip_addr/mask remote-ip ip_addr/mask svc_name
}
```

unset

```
unset vpn vpn_name
[ bind { interface | zone } | monitor | proxy-id ]
```

Keywords and Variables

Variable Parameters

```
get vpn tunn_str [ ... ]
```

name_str Defines a name for the VPN.

Example: The following command displays a VPN named “branch”:

```
get vpn branch
```

ah

```
set vpn tunn_str manual spi_num1 spi_num2 gateway ip_addr [ ... ] ah { ... }
```

ah Specifies Authentication Header (AH) protocol to authenticate IP packet content.

- **md5** Specifies the Message Digest 5 (MD5) hashing algorithm. (128-bit)
- **sha-1** Specifies the Secure Hash Algorithm (version) 1 (SHA-1) hashing algorithm. (160-bit)

The **key** *key_str* value defines a 16-byte (MD5) or 20-byte (SHA-1) hexadecimal key, which the NetScreen device uses to produce a 96-bit message digest (or hash) from the message.

password *pswd_str* Specifies a password the NetScreen device uses to generate an encryption or authentication key automatically.

Example: The following command creates a Manual Key VPN tunnel named “Mkt_vpn”.

- Sets the local and remote SPI values as 2002 and 3003
- Defines the remote gateway address 2.2.2.2
- Specifies Authentication Header (AH) protocol for IP packet authentication using the SHA-1 algorithm, the key for which is generated from the password “swordfish”

```
set vpn Mkt_vpn manual 2002 3003 gateway 2.2.2.2 ah sha-1 password swordfish
```

auto

get vpn auto

auto Displays all AutoKey IKE VPNs.

Example: The following command displays all AutoKey IKE VPNs:

get vpn auto

bind

```
set vpn tunn_str bind { interface interface | zone name_str }  
unset vpn vpn_name bind { interface | zone }
```

bind Binds VPN tunnel to a tunnel interface or a security zone.

- **interface** *interface* specifies the tunnel interface to use for VPN binding.
- **zone** *name_str* specifies the tunnel zone to use for VPN binding.

Example: The following command binds the VPN tunnel named vpn1 to the tunnel.1 interface:

set vpn vpn1 bind interface tunnel.1

Example: The following command binds the VPN tunnel named vpn2 to the Untrust-Tun tunnel zone:

set vpn vpn2 bind zone untrust-tun

df-bit

```
set vpn tunn_str df-bit { clear | copy | set }
```

- df-bit** Determines how the NetScreen device handles the Don't Fragment (DF) bit in the outer header.
- **clear** Clears (disables) DF bit from the outer header. This is the default value.
 - **copy** Copies the DF bit to the outer header.
 - **set** Sets (enables) the DF bit in the outer header.

esp

```
set vpn tunn_str manual spi_num1 spi_num2 gateway ip_addr esp { ... }
```

- esp** Specifies the use of the Encapsulating Security Payload (ESP) protocol, which the NetScreen device uses to encrypt and authenticate IP packets.
- **aes128** Specifies Advanced Encryption Standard (AES). The **key** *key_str* value defines a 128-bit hexadecimal key.
 - **aes192** Specifies Advanced Encryption Standard (AES). The **key** *key_str* value defines a 192-bit hexadecimal key.
 - **aes256** Specifies Advanced Encryption Standard (AES). The **key** *key_str* value defines a 256-bit hexadecimal key.
 - **des** Specifies Data Encryption Standard (DES). The **key** *key_str* value defines a 64-bit hexadecimal key (truncated to 56 bits).
 - **3des** Specifies Triple Data Encryption Standard (3DES). The **key** *key_str* value defines a 192-bit hexadecimal key (truncated to 168 bits).
 - **null** Specifies no encryption. (When you specify this option, you must specify an authentication algorithm (MD5 or SHA-1) using the **auth** option.)

auth Specifies the use of an authentication (hashing) method. The available choices are MD5 or SHA-1. (Some NetScreen devices do not support SHA-1.) The **key** *key_str* value defines a 16-byte (MD5) or 20-byte (SHA-1) hexadecimal key, which the NetScreen device uses to produce a 96-bit message digest (or hash) from the message.

password *pswd_str* Specifies a password the NetScreen device uses to generate an encryption or authentication key automatically.

Example: The following command creates a Manual Key VPN tunnel named “Mkt_vpn”.

- Specifies local and remote SPI values 2002 and 3003
- Specifies the IP address of the remote gateway 2.2.2.2
- Specifies ESP with 3DES encryption and SHA-1 authentication
- Generates the encryption and authentication keys from the passwords “swordfish” and “avalanche”

```
set vpn Mkt_vpn manual 2002 3003 gateway 2.2.2.2 esp 3des password swordfish auth  
sha-1 password avalanche
```

failover-weight

```
set vpn name_str failover-weight number }
```

failover-weight Assigns a weight to a VPN tunnel. When the accumulated weight of failed or “down” VPN tunnels bound to the primary Untrust zone interface reaches or exceeds 100%, ScreenOS fails over to the backup Untrust zone interface.

Note: This option is available only on devices that support the DIAL-backup feature.

Example: The following command assigns a failover weight of 50% to the VPN to_remote1:

```
set vpn to_remote1 failover-weight 50
```

gateway

```
set vpn tunn_str gateway ip_addr [ ... ] { ... }
set vpn tunn_str gateway name_str [ ... ] { ... }
get vpn gateway [ detail ]
```

- gateway** Specifies the autokey IKE gateway (*ip_addr* or *name_str*) to use.
- **idletime** *number* The length of time in minutes that a connection can remain inactive before the NetScreen device terminates it.
 - **replay | no-replay** Enables or disables replay protection. The default setting is no-replay.
 - **transport | tunnel** Defines the IPSec mode. In tunnel mode, the active IP packet is encapsulated. In transport mode, no encapsulation occurs. Tunnel mode is appropriate when both of end points in an exchange lie beyond gateway devices. Transport mode is appropriate when either end point is a gateway.
 - **proposal** *name_str* Defines up to four Phase 2 proposals. A Phase 2 proposal determines how a NetScreen device sends VPN session traffic.
 - **sec_level** Specifies a predefined set of proposals.

Example: In the following example you define an IKE gateway for a remote site in London. The gateway has the following elements:

- The remote gateway is named London_Office, with IP address 2.2.2.2.
- The outgoing interface is ethernet3.
- The Phase 1 proposal consists of the following components:
 - DSA certificate for data source authentication
 - Diffie-Hellman group 2 to protect the exchange of keying information
 - AES-128 encryption algorithm
 - MD-5 authentication algorithm

You then reference that gateway in a VPN tunnel that has the following elements:

- The tunnel is named London_Tunnel.
- The Phase 2 proposal consists of the following components:
 - Diffie-Hellman group 2 to protect the keying information during Phase 2 key exchanges
 - Encapsulating Security Payload (ESP) to provide both confidentiality through encryption and encapsulation of the original IP packet and integrity through authentication
 - AES-128 encryption algorithm
 - MD-5 authentication algorithm

```
set ike gateway London_Office ip 2.2.2.2 outgoing-interface ethernet3 proposal  
  dsa-g2-aes128-md5  
set vpn London_Tunnel gateway London_Office proposal g2-esp-aes128-sha
```

manual

```
get vpn tunn_str [ detail ] manual  
set vpn tunn_str manual spi_num1 spi_num2 gateway ip_addr [ ... ] { ... }
```

manual Specifies a Manual Key VPN. When the NetScreen device is in Manual mode, you can encrypt and authenticate by HEX key or password.

spi_num1 and *spi_num2* are 32-bit *local* and *remote* security parameters index (SPI) numbers. Each SPI number uniquely distinguishes a particular tunnel from any other active tunnel. Each must be a hexadecimal value between 3000 and 2ffffff.

The local SPI corresponds to the remote SPI at the other end of the tunnel, and vice-versa.

monitor

```
set vpn tunn_str monitor [ ... ] [ destination-ip ip_addr ] [ ... ]  
unset vpn tunn_str monitor
```

- monitor** Directs the NetScreen device to send VPN monitor messages to a NetScreen-Remote client or a non-NetScreen peer device.
- The **source-interface** *interface* option specifies the interface through which the NetScreen device sends the monitor messages.
- **destination-ip** specifies the destination IP address for the VPN monitoring feature to ping.
 - **optimized** performs optimization for scalability.
 - **rekey** triggers rekey of an autokey VPN if a tunnel is down.

Example: The following command uses ethernet3 as the source interface and 10.1.1.5 as the destination IP address for VPN monitoring through a VPN tunnel named tun1:

```
set vpn tun1 monitor source-interface ethernet3 destination-ip 10.1.1.5
```

outgoing-interface

```
set vpn tunn_str manual spi_num1 spi_num2 gateway ip_addr [ ... ]  
    outgoing-interface interface { ... }
```

- outgoing-interface** Defines the interface through which the NetScreen device sends traffic for this Manual Key VPN.
- For more information on interfaces, see [“Interface Names” on page A-1](#).

proxy-id

```
get vpn proxy-id
set vpn tunn_str proxy-id local-ip ip_addr/mask remote-ip ip_addr/mask svc_name
unset vpn vpn_name proxy-id
```

proxy-id Specifies the three-part tuple consisting of local IP address–remote IP address–service.

- **local-ip** *ip_addr/mask* The local IP address that sends and receives traffic through the tunnel.
- **remote-ip** *ip_addr/mask* The remote IP address that sends and receives traffic through the tunnel.
- *svc_name* The name of the service, such as FTP, TELNET, DNS or HTTP that passes through the tunnel. (Specifying **any** enables all services.)

Example: The following command creates a VPN proxy configuration for a VPN (**Sales**) with the HTTP service:

```
set vpn Sales proxy-id local-ip 10.1.1.0/24 remote-ip 10.2.2.0/24 HTTP
```

rekey

```
set vpn corp monitor rekey
```

rekey Keeps the SA active even if there is no other VPN traffic.

sec-level

```
set vpn tunn_str gateway { name_str | ip_addr } [ ... ] { ... } sec-level
{ basic | compatible | standard }
```

sec-level Specifies which pre-defined security proposal to use for IKE. The **basic** proposal provides basic-level security settings. The **compatible** proposal provides the most widely-used settings. The **standard** proposal provides settings recommended by Juniper Networks.

vpn-group

Description: Use the **vpn-group** commands to define or remove VPN groups, or to display VPN groups.

A *VPN group* is a collection of defined VPN tunnels. A VPN group allows the NetScreen device to perform tunnel failover. Each tunnel in the group has an assigned weight. When the NetScreen device invokes a policy that uses a VPN group, the device constructs all tunnels in the group, and the tunnel with the greatest weight becomes active by default. The IKE heartbeat periodically checks to see if this tunnel is working. If it is not, the device uses the tunnel with the next highest weight.

Syntax

get

```
get vpn-group [ id id_num ]
```

set

```
set vpn-group id id_num [ vpn tunn_str [ weight number ] ]
```

unset

```
unset vpn-group id id_num [ vpn tunn_str [ weight number ] ]
```

Keywords and Variables

id

```
get vpn-group id id_num  
set vpn-group id id_num [ ... ]  
unset vpn-group id id_num [ ... ]
```

id Specifies an identification number for the VPN group.

vpn

```
set vpn-group id id_num vpn tunn_str [ ... ]  
unset vpn-group id id_num vpn tunn_str
```

vpn Specifies the name of a VPN to place in the VPN group.

weight

```
set vpn-group id id_num vpn tunn_str weight number  
unset vpn-group id id_num vpn tunn_str weight number
```

weight Specifies a weight (priority) for the VPN relative to other VPNs in the group. The higher the number, the higher priority.

Example: With the following commands, you create two VPN tunnels (vpn1 and vpn2). You place them in a VPN group with ID 1001, which you then reference in a policy permitting traffic from addr1 in the Trust zone to addr2 in the Untrust zone beyond the remote gateway. You assign vpn1 a greater weight, giving it priority. If traffic cannot pass through vpn1, the NetScreen device redirects it through vpn2:

```
set ike gateway gw1 ip 1.1.1.1 preshare bi273T1L proposal pre-g2-3des-md5  
set ike gateway gw2 ip 2.2.2.2 preshare r3ix6403 proposal pre-g2-aes128-md5  
set vpn vpn1 gateway gw1 replay proposal g2-esp-3des-sha  
set vpn vpn2 gateway gw2 replay proposal g2-esp-3des-sha  
set vpn-group id 1001 vpn vpn1 weight 1  
set vpn-group id 1001 vpn vpn2 weight 2  
set policy from trust to untrust addr1 addr2 HTTP tunnel vpn-group 1001
```

vpnmonitor

Description: Use the **vpnmonitor** commands to set the monitor frequency and threshold.

The NetScreen ScreenOS provides the ability to determine the status and condition of active VPNs through the use of ICMP pings, and report the conditions by using SNMP VPN monitoring objects and traps.

To enable your SNMP manager application to recognize the VPN monitoring MIBs, you must import the NetScreen-specific MIB extension files into the application. The MIB extension files are on the Juniper Networks NetScreen documentation CD that shipped with the NetScreen device.

Syntax

get

```
get vpnmonitor
```

set

```
set vpnmonitor  
{  
  interval number |  
  threshold number  
}
```

unset

```
unset vpnmonitor interval  
{  
  interval |  
  threshold  
}
```


Keywords and Variables

interval

```
set vpnmonitor interval number  
unset vpnmonitor interval
```

interval Specifies the monitor frequency interval (in seconds).

threshold

```
set vpnmonitor threshold number  
unset vpnmonitor threshold
```

threshold Specifies the monitor threshold, the number of consecutive times the device can send vpnmonitor requests without getting a response before the device changes the VPN Link-Status to down.

vrouter

Description: Use the **vrouter** commands to configure a virtual router on the NetScreen device.

Executing the **set vrouter *name_str*** command without specifying further options places the CLI in the routing context. For example, the following command places the CLI in the **trust-vr** routing context:

```
set vrouter trust-vr
```

Once you initiate the routing context, all subsequent command executions apply to the specified local virtual router (**trust-vr** in this example). You can then initiate the **bgp**, **ospf**, or **rip** protocol context.

- To enter the **bgp** context, execute the **set protocol bgp** command.
`ns(trust-vr)-> set protocol bgp`
- To enter the **ospf** context, execute the **set protocol ospf** command.
`ns(trust-vr)-> set protocol ospf`
- To enter the **rip** context, execute the **set protocol rip** command.
`ns(trust-vr)-> set protocol rip`

In the **bgp**, **ospf**, or **rip** protocol context, all command executions apply to the specified protocol.

Syntax

exec

```
exec vrouter name_str protocol bgp neighbor ip_addr
{
  connect |
  disconnect |
  tcp-connect
}
```

get

```
get vrouter name_str
[
  access-list |
  config |
  default-vrouter |
  interface |
  preference |
  protocol { bgp | ospf | rip }1 |
  route
  [
    id id_num |
    ip ip_addr |
    prefix ip_addr/mask |
    protocol { bgp | connected | imported | ospf | rip | static }
    source [ id id_num | ip ip_addr | prefix ip_addr/mask ] |
    summary
  ] |
  route-map [ name_str ]
  [
    config |
    number [ config | match | set ]
  ] |
  router-id |
  rule |
  statistics |
  zone
]
```

1. For more information on the **protocol { bgp | ospf | rip }** options, see the **bgp**, **ospf**, and **rip** command descriptions.

set

```
set vrouter { name name_str | name_str }
[
  access-list id_num
    { permit | deny } { ip ip_addr/mask | default-route } number |
  add-default-route vrouter untrust-vr |
  adv-inact-interface
  auto-route-export |
  default-vrouter |
  enable-source-routing
  export-to | import-from
    vrouter name_str route-map name_str protocol
      { bgp | connected | imported | ospf | rip | static }
  ignore-subnet-conflict |
  max-routes number |
  nsrp-config-sync |
  preference
    {
      auto-exported number |
      connected number |
      ebgp number |
      ibgp number |
      imported number |
      ospf number |
      ospf-e2 number |
      rip number
      static number
    } |
  protocol { bgp | ospf | rip }2 |
  route [ source ] ip_addr/mask
    {
      interface interface
```

2. For more information on the **protocol** options, see the **bgp**, **ospf**, or **rip** command descriptions.

```

        [ gateway ip_addr ] [ metric number ] [ tag id_num ] |
vrouter name_str
} |
route-map
{
name name_str { permit | deny } number |
name_str number }
[
as-path id_num |
community id_num |
local-pref number |
match
{
as-path id_num |
community id_num |
interface interface |
ip id_num |
metric number |
next-hop id_num |
route-type
{ internal-ospf | type1-external-ospf | type2-external-ospf } |
tag { number | ip_addr }
}|
metric number |
metric-type { type-1 | type-2 } |
next-hop ip_addr |
offset-metric number |
tag { number | ip_addr } |
weight number
]
router-id { id_num | ip_addr } |
sharable
snmp trap private
]

```

unset

```
unset vrouter name_str
[
  access-list id_num [ ip_addr/mask number | default-route number ] |
  add-default-route |
  adv-inact-interface |
  auto-route-export |
  enable-source-routing |
  export-to | import-from
    vrouter name_str route-map name_str protocol
      { bgp | connected | imported | ospf | rip | static } |
  ignore-subnet-conflict |
  max-routes |
  nsrp-config-sync |
  preference
    {
      auto-exported |
      connected |
      ebgp |
      ibgp |
      imported |
      ospf |
      ospf-e2 |
      rip |
      static
    } |
  protocol { bgp | ospf | rip }3 |
  route [ source ] ip_addr/mask
    [
      vrouter name_str |
      interface interface [ gateway ip_addr ]
    ] |
```

3. For more information on the **protocol { bgp | ospf | rip }** options, see the **bgp**, **ospf**, and **rip** command descriptions.

```
route-map name_str
[ number ] |
as-path |
community |
local-pref |
match
{
as-path |
community |
interface |
ip |
metric |
next-hop |
route-type |
tag |
}|
metric |
metric-type |
next-hop |
offset-metric |
tag |
weight
] |
router-id |
sharable
snmp trap private
]
```

Keywords and Variables

Variable Parameter

set vrouter *name_str*

name_str The name of the virtual router. The name can be a predefined virtual router, such as trust-vr or untrust-vr, or it can be a user-defined virtual router created with the **name** keyword. (Creating custom virtual routers is only supported on certain NetScreen devices and requires a vsys software key.).

Example: The following commands activate the trust-vr virtual router context, activate the BGP routing context, and execute the context-dependent command **get config**.

```
set vrouter trust-vr
ns(trust-vr)-> set protocol bgp
ns(trust-vr/bgp)-> get config
```

access-list

```
get vrouter name_str access-list
set vrouter name_str access-list id_num
    { permit | deny } { ip ip_addr/mask | default-route } number }
unset vrouter name_str access-list id_num [ ip_addr/mask | default-route ] number
```

access-list Creates or removes an access list, or entries in the access list. Each entry permits (or denies) routes according to IP address and mask, or default route. The *id_num* value identifies the access list. The *number* identifies the sequence number for this entry in the access list.

- **permit** Directs the virtual router to permit the route.
- **deny** Directs the virtual router to deny the route.
- **default-route** Enters the default route for the virtual router into the access list.

add-default-route

```
set vrouter name_str add-default-route vrouter name_str  
unset vrouter name_str add-default-route
```

add-default-route Adds a default route with the next hop as another virtual router. (This command is available only in the default virtual router of the current vsys, and only if this virtual router is not untrust-vr.)

adv-inact-interface

```
set vrouter name_str adv-inact-interface  
unset vrouter name_str adv-inact-interface
```

adv-inact-interface Directs the virtual router to consider active routes on inactive interfaces for redistribution or export. By default, only active routes defined on active interfaces can be redistributed to other protocols or exported to other virtual routers.

auto-route-export

```
set vrouter name_str auto-route-export  
unset vrouter name_str auto-route-export
```

auto-route-export Directs the virtual router to export public interface routes to the untrust-vr vrouter. An interface is public if it is in Route mode, and private if it is in NAT mode. For information on Route mode and NAT mode, refer to the *NetScreen Concepts and Examples Reference Guide*.

Note: The auto-route-export switch does not take effect if the specified vrouter (*name_str*) has export or import rules to the untrust-vr virtual router.

config

get vrouter *name_str* **config**

config Displays configuration information about the virtual router.

default-vrouter

get vrouter *name_str* **default-vrouter**
set vrouter *name_str* **default-vrouter**

default-vrouter Sets the specified virtual router as the default router for the vsys.

enable-source-routing

get vrouter *name_str* **enable-source-routing**
set vrouter *name_str* **enable-source-routing**

enable-source-routing Directs the virtual router to perform routing table lookups based on source IP address.

export-to | import-from

set vrouter *name_str* { **export-to** | **import-from** } **vrouter** *name_str* { ... }
unset vrouter *name_str* { **export-to** | **import-from** } **vrouter** *name_str* { ... }

export-to | import-from Directs the virtual router to import routes from another virtual router (source), or to export routes to another virtual router (destination).

- **vrouter** *name_str* identifies the source or destination virtual router.
- **route-map** *name_str* identifies the route map that filters the imported or exported routes.
- **protocol** Specifies the protocol for the imported or exported routes.
 - **bgp** Directs the virtual router to import or export Border Gateway Protocol (BGP) routes.
 - **connected** Directs the virtual router to import or export connected routes.

- **imported** Directs the virtual router to import or export routes that were redistributed into the virtual router from another virtual router.
- **ospf** Directs the virtual router to import or export Open Shortest Path First (OSPF) routes.
- **rip** Directs the virtual router to import or export Routing Information Protocol (RIP) routes.
- **static** Directs the virtual router to import or export static routes.
- **default-route** Directs the virtual router to export or import the default route.

ignore-subnet-conflict

```
set vrouter name_str ignore-subnet-conflict
unset vrouter name_str ignore-subnet-conflict
```

ignore-subnet-conflict Directs the virtual router to ignore overlapping subnet addresses for interfaces in the virtual router. By default, you cannot configure overlapping subnet IP addresses on interfaces in the same virtual router.

interface

```
get vrouter name_str interface
```

interface Displays the interfaces in the virtual router.

max-routes

```
set vrouter name_str max-routes number
unset vrouter name_str max-routes
```

max-routes Specifies the maximum number of routing entries allowed for this virtual router. By default, the maximum number of entries allowed for a virtual router depends upon the NetScreen device and the number of virtual routers configured on the device.

name

set vrouter name *name_str*

name Specifies the name of a user-defined virtual router. Creating custom virtual routers is only supported on certain NetScreen devices and requires a vsys software key.

nsrp-config-sync

set vrouter *name_str* **nsrp-config-sync**
unset vrouter *name_str* **nsrp-config-sync**

nsrp-config-sync Synchronizes the specified virtual router (*name_str*) with the same virtual router on a NSRP peer. This switch is enabled by default.

preference

get vrouter *name_str* **preference**
set vrouter *name_str* **preference**
unset vrouter *name_str* **preference**

preference Specifies route preference level based upon protocol. The lower the value, the more preference given to the route. You can specify a value between 1-255.

- **auto-exported** Specifies preference levels for routes (defined on public interfaces) that the virtual router automatically exports to the untrust-vr virtual router. The default is 30.
- **connected** Specifies preference level for connected routes. The default is 0.
- **ebgp** Specifies preference level for External Border Gateway Protocol (EBGP) routes. The default is 120.
- **ibgp** Specifies preference level for Internal Border Gateway Protocol (IBGP) routes. The default is 40.

- **imported** Specifies preference level for pre-existing routes exported to another protocol and passed on to other routers. The default is 140.
- **ospf** Specifies preference level for Open Shortest Path First (OSPF) routes. The default is 60.
- **ospf-e2** Specifies preference level for OSPF External Type 2 routes. The default is 200.
- **rip** Specifies preference level for Routing Information Protocol (RIP) routes. The default is 100.
- **static** Specifies preference level for static routes. The default is 20.

protocol

```
exec vrouter name_str protocol { ... }  
get vrouter name_str protocol { bgp | ospf | rip }  
set vrouter name_str protocol { bgp | ospf | rip }  
unset vrouter name_str protocol { bgp | ospf | rip }
```

protocol Places the NetScreen device in the BGP context or the OSPF context. (For information on these contexts, see the **bgp**, **ospf**, or **rip** command descriptions in this manual.)

The **exec vrouter name_str protocol bgp neighbor ip_addr** command has the following options:

- **connect** Establishes a BGP connection to the specified neighbor.
- **disconnect** Terminates a BGP connection to the specified neighbor.
- **tcp-connect** Tests the TCP connection to the neighbor.

route

```
get vrouter name_str route [ ... ]  
set vrouter name_str route [ source ] ip_addr/mask  
{  
  interface interface [ gateway ip_addr [ metric number ] [ tag id_num ] ] |  
  vrouter name_str  
}
```

```
unset vrouter name_str route [ source ] ip_addr/mask
[
  vrouter name_str |
  interface interface [ gateway ip_addr ]
]
```

route

Configures routes for the routing table for the virtual router.

- *ip_addr/mask* Specifies the IP address that appears as an entry in the routing table.
- **gateway** *ip_addr* Specifies the gateway for the next hop. The **metric** value specifies the cost of the route.
- **id** *id_num* Displays information for the route that matches the ID number. The ID number is a system-assigned number that you can see when you enter the **get vrouter name_str route** command with no options.
- **interface** *interface* Specifies the routed interface.
- **ip** *ip_addr* Displays the route for the specified IP address.
- **prefix** *ip_addr/mask* Displays the routes within the specified subnet address.
- **protocol** Displays BGP, connected, imported, OSPF, RIP, or static routes.
- **source** Specifies that the route is a source-based route. When displaying a source-based route, you can optionally specify:
 - **id** *id_num*
 - **ip** *ip_addr*
 - **prefix** *ip_addr/mask*
- **summary** Displays a summary of the routes.
- **vrouter** *name_str* Specifies a virtual router as the next hop.

route-map

```
get vrouter name_str route-map [ ... ]
set vrouter name_str { ... } vrouter name_str route-map
  { name name_str | name_str } [ ... ]
unset vrouter name_str { ... } vrouter name_str route-map name_str [ ... ]
```

route-map

Configures a route map for the virtual router.

With the **name** keyword, the **route-map** option creates a new route map (*name_str*). Otherwise, *name_str* configures an existing route map. Each entry in the route map must have a sequence number (*number*) that identifies the order in which the route map entries are compared against an incoming or outgoing route. The **permit** and **deny** switches determine if the entry allows redistribution of routes to another virtual router or another protocol.

The **match** keyword directs the virtual router to match routes to specified parameters. You can match the following parameters:

- **as-path** *id_num* Specifies an AS path access list that defines the BGP AS path attribute to be matched.
- **community** *id_num* Specifies a BGP community list (*id_num*) that defines the community attribute to be matched.
- **interface** *interface* Specifies an interface on the NetScreen device.
- **ip** *id_num* Specifies an access list that defines the IP addresses of routes to be matched.
- **metric** *number* The cost of the route. Enter a number between 1-65535.
- **next-hop** *id_num* Specifies an access list that defines the next-hop for routes to be matched
- **route-type** Specifies which kind of OSPF route matches the route map entry.
 - **internal-ospf** Matches only OSPF internal routes.
 - **type1-external-ospf** Matches only external OSPF Type-1 routes.
 - **type2-external-ospf** Matches only external OSPF Type-2 routes.
- **tag** { *number* | *ip_addr* } Matches either a route tag or an IP address.

Other keywords allow you to optionally set values for parameters on matching routes. You can set the following parameters:

- **as-path** *id_num* Specifies the AS path access list values that are prepended to the path list of the matching route.
- **community** *id_num* Specifies the community list values that are set in the community attribute for the matching route.
- **local-pref** *number* Specifies the path preference for the matching route.
- **metric** *number* Specifies the metric for the matching route. Enter a number between 1-65535.
- **metric-type** Specifies OSPF metric type that is set for the matching route.
 - **type-1** Specifies OSPF Type-1 route.
 - **type-2** Specifies OSPF Type-2 route.
- **next-hop** *ip_addr* Specifies the next hop IP address for the matching route.
- **offset-metric** *number* Specifies the RIP offset to increase the metric for the matching route.
- **tag** { *number* | *ip_addr* } Specifies a tag or IP address for the matching route.
- **weight** *number* Sets the weight of the matching route for BGP.

While configuring a route map, you can use the get **config**, **get match**, and **get set** commands to display route map configuration commands, or match or set conditions.

router-id

```
get vrouter name_str router-id  
set vrouter name_str router-id { id_num | ip_addr }  
unset vrouter name_str router-id
```

router-id Specifies the router identification that the virtual router uses to communicate with other routing devices. You can enter the router identification in either a dotted decimal notation (like an IP address) or a decimal number (this is converted to 0.0.0.*number*). If you do not specify a router identification, the device uses the highest IP address of the any interface in the virtual router as the router identification.

rule

```
get vrouter name_str rule
```

rule Displays import and export rules for the virtual router.

sharable

```
set vrouter name_str sharable  
unset vrouter name_str sharable
```

sharable Makes the root-level virtual router accessible from any virtual system (vsys) on the device.

snmp

```
set vrouter name_str snmp trap private  
unset vrouter name_str snmp trap private
```

snmp Makes SNMP traps private for the dynamic routing MIBs under the virtual router. Private traps include the virtual router identification. This option is available only for the default root-level virtual router. (This is usually the trust-vr virtual router, although you can change the default virtual router at the root level.)

statistics

```
get vrouter name_str statistics
```

statistics Displays statistics for the virtual router.

zone

get vrouter *name_str* **zone**

zone Displays the zones bound to the virtual router.

vsys

Description: Use the **vsys** commands to create and configure virtual systems from the root level of a NetScreen device.

Virtual systems allow you to logically partition a single NetScreen security system to provide multi-tenant services. Each virtual system (vsys) is a unique security domain and can have its own administrators, called “virtual system administrators” or “vsys admins”. Such administrators can individualize their security domain by setting their own address books, virtual routers, user lists, custom services, VPNs, and policies. (Only a root-level administrator can set firewall security options, create virtual system administrators, and define interfaces and subinterfaces.)

When you execute the **set vsys** command, the command prompt changes to indicate that you are now operating within a virtual system. Use the **unset vsys** command to remove a specific virtual system and all its settings.

Syntax

get

```
get vsys [ name_str ]
```

set

```
set vsys name_str
[
  vrouter
  [
    name [ name_str ] [ id id_num ] [ vsd number ] |
    share [ name_str ] [ vsd number ] |
    vsd number
  ] |
  vsd number
]
```

unset

```
unset vsys name_str
```

Keywords and Variables

Variable Parameters

```
get vsys [ name_str ]  
set vsys name_str  
unset vsys name_str
```

name_str Defines the name of a virtual system and automatically places the root level admin within the virtual system. Subsequent commands configure the newly created virtual system.

Example: The following command creates a virtual system named **vsys1** and switches the console to the new virtual system:

```
set vsys vsys1
```

vrouter

```
set vsys name_str vrouter [ name [ name_str ] [ id id_num ] [ vsd number ] ]  
set vsys name_str vrouter [ share [ name_str ] [ vsd number ] ]
```

vrouter Defines and configures the default virtual router for the vsys.

- **name** Specifies a name for the virtual router.
 - **id id_num** Assigns an identification number to the virtual router.
 - **vsd id_num** See [“vsd” on page 543](#).
- **share** Specifies a shared root-level virtual router to use as a default router.
- **vsd id_num** See [“vsd” on page 543](#).

Examples: The following command creates a virtual system named Acme_Org, creates a virtual router named Acme_Router with ID 1025, and switches the console to the new virtual system:

```
set vsys Acme_Org vrouter name Acme_Router id 1025
```

The following command creates a virtual system named Acme_Org, and specifies a default, root-level virtual router (trust-vr):

```
set vsys Acme_Org vrouter share trust-vr
```

vsd

```
set vsys name_str vrouter [ vsd number ]
```

vsd number Assigns a Virtual Security Device (VSD) group number to the virtual router. A VSD group is a pair of physical NetScreen devices (a master and a backup) that collectively comprise a single VSD. A VSD provides failover capability, allowing the backup device to take over if the master device fails. For more information on VSD groups, see the *NetScreen Concepts & Examples ScreenOS Reference Guide*.

Example: The following command creates a virtual system named Acme_Org, creates a virtual router named Acme_Router, creates a VSD ID 5, and switches the console to the new virtual system:

```
set vsys Acme_Org vrouter vsd 5
```

webauth

Description: Use the **webauth** commands to configure the NetScreen device to perform WebAuth authentication.

WebAuth is an authentication method that requires the user to initiate an HTTP session and input authentication information, before the user can send traffic to the destination node.

You specify authentication in policy definitions (see the **auth** keyword description in [“policy” on page 376](#)).

Syntax

get

```
get webauth [ banner ]
```

set

```
set webauth { banner success string | server name_str }
```

unset

```
unset webauth { banner success | server }
```

Keywords and Variables

banner success

```
get webauth banner
set webauth banner success string
unset webauth banner success
```

banner success Specifies the banner (*string*) displayed in response to Webauth success.

Example: The following command changes the Webauth success banner to “Webauth service successful”:

```
set webauth banner success "Webauth service successful"
```

server

```
set webauth server name_str
unset webauth banner server
```

server Specifies the Webauth server name (*name_str*). (You can obtain all existing Webauth server names by executing the command **get auth-server all**.)

Example: The following command specifies a Webauth server named “Our_Webauth”:

```
set webauth server Our_Webauth
```

Defaults

The default banner value is **Webauth Success**.

webtrends

Description: Use the **webtrends** commands to configure the NetScreen device for WebTrends.

The WebTrends Firewall Suite allows you to customize syslog reports of critical, alert, and emergency events to display the information you want in a graphical format. You can create reports that focus on areas such as firewall attacks (emergency-level events) or on all events with the severity levels of critical, alert, and emergency.

Syntax

get

```
get webtrends
```

set

```
set webtrends
{
  VPN |
  enable |
  host-name name_str |
  port port_num
}
```

unset

```
unset webtrends
{
  VPN |
  enable |
  host-name |
  port
}
```


Keywords and Variables

vpn

```
set webtrends VPN
unset webtrends VPN
```

vpn Enables WebTrends VPN encryption.

enable

```
set webtrends enable
unset webtrends enable
```

enable Enables WebTrends.

host-name

```
set webtrends host-name name_str
unset webtrends host-name
```

host-name Specifies the WebTrends host name.

port

```
set webtrends port port_num
unset webtrends port
```

port *port_num* Specifies the WebTrends host port.

Description: Use the **xauth** commands to configure the NetScreen device to perform XAuth authentication.

An XAuth user or user group is one or more remote users who authenticate themselves when connecting to the NetScreen device via an AutoKey IKE VPN tunnel and optionally receive TCP/IP settings from the NetScreen device. Whereas IKE user authentication is actually the authentication of VPN gateways or clients, XAuth user authentication is the authentication of the users themselves. It requires each user to enter information unique to that user (the user name and password).

Syntax

get

```
get xauth { active | default | lifetime }
```

set

```
set xauth
{
  default
  {
    auth server name_str [ chap ] [ query-config ] |
    dns1 ip_addr | dns2 ip_addr |
    ippool name_str |
    wins1 ip_addr | wins2 ip_addr
  } |
  lifetime number
}
```

unset

```
unset xauth
{
  default { dns1 | dns2 | ippool | wins1 | wins2 } |
  lifetime
}
```

Keywords and Variables

active

```
get xauth active
```

active Displays all currently active XAuth login instances.

default

```
get xauth default
set xauth default { ... }
unset xauth default { ... }
```

default Sets or displays default XAuth settings.

- **auth server** Identifies the XAuth server by object name (*name_str*).
 - **chap** Directs the NetScreen to use Challenge Handshake Authentication Protocol (CHAP) while performing authentication with the XAuth client.
 - **query-config** Sets query client settings (such as IP address) from the external authentication server.
- **dns1** Identifies the DNS primary server by ip address (*ip_addr*).
- **dns2** Identifies the DNS secondary server by ip address (*ip_addr*).
- **ippool** Identifies the IP pool (*name_str*).

- **wins1** Identifies the WINS primary server by ip address (*ip_addr*).
- **wins2** Identifies the WINS secondary server by ip address (*ip_addr*).

Example: The following command sets up the NetScreen device to use a XAuth server (Our_Auth):

```
set xauth default auth server Our_Auth
```

lifetime

```
get xauth lifetime
set xauth lifetime number
unset xauth lifetime number
```

lifetime *number* Specifies the maximum length of time (in minutes) that the XAuth server holds resources (such as IP address) on behalf of the client.

Example: The following command specifies a maximum XAuth session length of 30 minutes:

```
set xauth lifetime 30
```

zone

Description: Use the **zone** commands to create, remove, or display a security zone, and to set screen options.

A *security zone* is method for sectioning the network into segments to which you can apply various security options. You can configure multiple security zones for individual NetScreen devices, thus dividing the network into segments to which you can apply security options. There must be at least two security zones per device, basically to protect one area of the network from the other. On some NetScreen platforms, you can define many security zones, bringing finer granularity to your network security design, without deploying multiple security appliances.

Each security zone has at least one interface bound to it. For a brief description of the interfaces, see [“Interface Names” on page A-I](#). For information on security zones, see [“Zone Names” on page B-I](#).

Syntax

get

```
get zone
  [
    id id_num |
    all |
    zone [ screen { all | attack | counter | info } ]
  ]
```

set

```
set zone
  {
    name zone [ { L2 id_num | tunnel zone } ] |
    zone
    {
      asymmetric vpn |
      block |
      screen
    }
  }
```

```
{
alarm-without-drop |
block-frag |
component-block [ activex | java | zip | exe ] |
fin-no-ack |
icmp-flood [ threshold number ] |
icmp-fragment |
icmp-large |
ip-bad-option |
ip-filter-src |
ip-loose-src-route |
ip-record-route |
ip-security-opt |
ip-spoofing [ drop-no-rpf-route ] |
ip-stream-opt |
ip-strict-src-route |
ip-sweep [ threshold number ] |
ip-timestamp-opt |
land |
limit-session
    [ source-ip-based number | destination-ip-based [ number ] ] |
mal-url { string1 string2 number | code-red } |
ping-death |
port-scan [ threshold number ] |
syn-ack-ack-proxy [ threshold number ] |
syn-fin |
syn-flood
    [
        alarm-threshold number |
        attack-threshold number |
        destination-threshold number |
        drop-unknown-mac |
        queue-size number |
        source-threshold number |
        timeout number
    ] |
}
```

```
        syn-frag |
        tcp-no-flag |
        tear-drop |
        udp-flood [ threshold number ] |
        unknown-protocol |
        winnuke
    }
    reassembly-for-alg |
    tcp-rst |
    vrouter name_str
} |
}
```

unset

```
unset zone zone
{
    asymmetric vpn |
    block |
    screen
    {
        block-frag |
        component-block |
        fin-no-ack |
        icmp-flood [ threshold ] |
        icmp-fragment |
        icmp-large |
        ip-bad-option |
        ip-filter-src |
        ip-loose-src-route |
        ip-record-route |
        ip-security-opt |
        ip-spoofing [ drop-no-rpf-route ] |
        ip-stream-opt |
        ip-strict-src-route |
    }
}
```

```
ip-sweep [ threshold ] |
ip-timestamp-opt |
land |
limit-session [ source-ip-based | destination-ip-based ] |
mal-url { string1 | code-red } |
ping-death |
port-scan [ threshold ] |
syn-ack-ack-proxy [ threshold number ] |
syn-fin |
syn-flood
    [
        alarm-threshold |
        attack-threshold |
        destination-threshold number |
        drop-unknown-mac |
        queue-size |
        source-threshold |
        timeout
    ] |
syn-frag |
tcp-no-flag |
tear-drop |
udp-flood [ threshold number ] |
unknown-protocol |
winnuke
}
tcp-rst |
}
```


Keywords and Variables

Variable Parameters

```
get zone zone [ ... ]  
set zone zone { ... }  
unset zone zone { ... }
```

zone The name of the zone. For more information on zones and zone names, see [“Zone Names” on page B-I](#).

all

```
get zone all [ ... ]
```

all Displays information on all existing zones.

asymmetric vpn

```
set zone asymmetric vpn
```

asymmetric vpn When enabled, this option allows any incoming VPN traffic in a zone to match any applicable VPN session, regardless of the origin for the original VPN tunnel. For example, traffic coming from VPN A can match a session created by traffic for VPN B. This feature allows free routing of VPN traffic between two or more sites when there are multiple possible paths for VPN traffic.

Note: It is not advisable to mix policy-based and route-based VPNs for asymmetric traffic.

block

```
set zone zone block  
unset zone zone block
```

block Imposes intra-zone traffic blocking.

name

```
set zone name zone { ... }
```

- name** Creates a new zone with name *zone*.
- **L2** *id_num* specifies that the zone is Layer-2 (for running the device in Transparent Mode). The ID number (*id_num*) identifies the VLAN to which the zone is bound. The name you specify (*zone*) must begin with "L2-".
 - **tunnel** *zone* specifies that the new zone is a VPN tunnel zone, and identifies the tunnel-out zone (*zone*).

Examples: The following command creates a new Layer-2 zone named **L2-Sales**, with VLAN ID number 1:

```
set zone name L2-Sales L2 1
```

The following command creates a tunnel zone named **Engineering**, and specify **untrust** as the out zone:

```
set zone name Engineering tunnel untrust
```

reassemble-for-alg

```
set zone untrust reassembly-for-alg
```

- reassemble-for-alg** Reassembles all fragmented IP packets and TCP segments for HTTP and FTP traffic that arrives at any interface bound to the zone on which you enable this option. With this option enabled, the NetScreen device can better detect malicious URLs that an attacker has deliberately broken into packet or segment fragments. Packet and segment reassembly also improves application layer gateway (ALG) filtering by allowing the NetScreen device to examine the complete text within payloads.

screen

```
set zone zone screen { ... }  
set zone zone screen { ... }
```

screen

Enables or disables firewall services through the interface.

- **alarm-without-drop** Generates an alarm when detecting an attack, but does not block the attack. This option is useful if you allow the attack to enter a segment of your network that you have previously prepared to receive it—such as a honeynet, which is essentially a decoy network with extensive monitoring capabilities.
- **block-frag** Enables IP packet fragmentation blocking.
- **component-block** Selectively blocks HTTP traffic containing any of the following components:
 - **activex** ActiveX controls
 - **java** Java applets
 - **exe** .EXE files
 - **zip** ZIP files

An attacker can use any of these components to load an application (a Trojan Horse) on a protected host, then use the application to gain control of the host. If you enable the blocking of HTTP components without specifying which components, the NetScreen device blocks them all. Alternatively, you can configure the NetScreen device to block only specified components.

Note: If you enable ActiveX-blocking, the NetScreen device also blocks packets containing Java applets, .exe files, and .zip files because they might be contained within an ActiveX control.

- **fin-no-ack** Detects an illegal combination of flags, and rejects packets that have them.
- **icmp-flood** [**threshold** *number*] Detects and prevents Internet Control Message Protocol (ICMP) floods. An ICMP flood occurs when ICMP echo requests are broadcast with the purpose of flooding a system with so much data that it first slows down, and then times out and is disconnected. The threshold defines the number of ICMP packets per second allowed to ping the same destination address before the NetScreen device rejects further ICMP packets. The range is 1 to 1,000,000.
- **icmp-fragment** Detects and drops any ICMP frame with the More Fragments flag set, or with an offset indicated in the offset field.

- **icmp-large** Detects and drops any ICMP frame with an IP length greater the 1024.
- **ip-bad-option** Detects and drops any packet with an incorrectly formatted IP option in the IP packet header. The NetScreen device records the event in the SCREEN counters list for the ingress interface.
- **ip-filter-src** Detects and drops all packets with the Source Route Option enabled. The Source Route Option can allow an attacker to use a false IP address to access a network, and receive returned traffic addressed to the real IP address of the attacker's host device. The administrator can block all IP Source Routed frames having Strict Source Routing (or Loose Source Routing) enabled.
- **ip-loose-src-route** Detects packets where the IP option is 3 (Loose Source Routing) and records the event in the SCREEN counters list for the ingress interface. This option specifies a partial route list for a packet to take on its journey from source to destination. The packet must proceed in the order of addresses specified, but it is allowed to pass through other routers in between those specified.
- **ip-record-route** Detects packets where the IP option is 7 (Record Route) and records the event in the SCREEN counters list for the ingress interface.
- **ip-security-opt** Detects packets where the IP option is 2 (security) and records the event in the SCREEN counters list for the ingress interface.
- **ip-spoofing** Prevents spoofing attacks. Spoofing attacks occur when unauthorized agents attempt to bypass firewall security by imitating valid client IP addresses. Using the **ip-spoofing** option invalidates such false source IP address connections. Only NetScreen devices running in NAT or Route mode can use this option. The **drop-no-rpf-route** option instructs the NetScreen device to drop any packet that is not contained in the route table, for example, the device drops the packet if it does not contain a source route, or if the source IP address is reserved (non-routable, as with 127.0.0.1).
- **ip-stream-opt** Detects packets where the IP option is 8 (Stream ID) and records the event in the SCREEN counters list for the ingress interface.
- **ip-strict-src-route** Detects packets where the IP option is 9 (Strict Source Routing) and records the event in the SCREEN counters list for the ingress interface. This option specifies the complete route list for a packet to take on its journey from source to destination. The last address in the list replaces the address in the destination field.

- **ip-sweep threshold** *number* Detects and prevents an IP Sweep attack. An IP Sweep attack occurs when an attacker sends ICMP echo requests (pings) to multiple destination addresses. If a target host replies, it reveals the target's IP address to the attacker. Set the IP Sweep threshold to between 1 and 1,000,000 microseconds. Each time ICMP echo requests occur with greater frequency than this limit, the NetScreen device drops further echo requests from the remote source address.
- **ip-timestamp-opt** Detects packets where the IP option list includes option 4 (Internet Timestamp) and records the event in the SCREEN counters list for the ingress interface.
- **land** Prevents Land attacks by combining the SYN flood defense mechanism with IP spoofing protection. Land attacks occur when an attacker sends spoofed IP packets with headers containing the target's IP address for both the source and destination IP addresses. The attacker sends these packets with the SYN flag set to any available port. This induces the target to create empty sessions with itself, filling its session table and overwhelming its resources.
- **limit-session** [**source-ip-based** *number* | **destination-ip-based** *number*] Limits the number of concurrent sessions the device can initiate from a single source IP address, or the number of sessions it can direct to a single destination IP address. By default, the limit is 128 sessions. Limit value range is 1 to 49,999.
- **mal-URL** [*name_str id_str number* | **code-red**] Sets up a filter that scans HTTP packets for suspect URLs. The NetScreen device drops packets that contain such URLs. The **code-red** switch enables blocking of the Code Red worm virus. Using the *name_str* option works as follows.
 - *name_str* A user-defined identification name.
 - *id_str* Specifies the starting pattern to search for in the HTTP packet. Typically, this starting pattern begins with the HTTP command GET, followed by at least one space, plus the beginning of a URL. (The NetScreen device treats multiple spaces between the command “GET” and the character “/” at the start of the URL as a single space.)
 - *number* Specifies a minimum length for the URL before the CR-LF.
- **ping-of-death** Detects and rejects oversized and irregular ICMP packets. Although the TCP/IP specification requires a specific packet size, many ping implementations allow larger packet sizes. This can trigger a range of adverse system reactions including crashing, freezing, and restarting.

- **port-scan threshold *number*** Prevents port scan attacks. A port scan attack occurs when an attacker sends packets with different port numbers to scan available services. The attack succeeds if a port responds. To prevent this attack, the NetScreen device internally logs the number of different ports scanned from a single remote source. For example, if a remote host scans 10 ports in 0.005 seconds (equivalent to 5000 microseconds, the default threshold setting), the NetScreen device flags this as a port scan attack, and rejects further packets from the remote source. The port-scan threshold *number* value determines the threshold setting, which can be from 1000 to 1,000,000 microseconds.
- **syn-ack-ack-proxy** Prevents the SYN ACK ACK attack. Such an attack occurs when the attacker establishes multiple Telnet sessions without allowing each session to terminate. This consumes all open slots, generating a Denial of Service condition.
- **syn-fin** Detects an illegal combination of flags attackers can use to consume sessions on the target device, thus resulting in a denial of service.
- **syn-flood** Detects and prevents SYN flood attacks. Such attacks occur when the connecting host continuously sends TCP SYN requests without replying to the corresponding ACK responses.
 - **alarm-threshold *number*** Defines the number of proxied, half-complete connections per second at which the NetScreen device makes entries in the event alarm log.
 - **attack_threshold *number*** Defines the number of SYN packets per second required to trigger the SYN proxying mechanism.
 - **destination-threshold *number*** Specifies the number of SYN segments received per second for a single destination IP address before the NetScreen device begins dropping connection requests to that destination. If a protected host runs multiple services, you might want to set a threshold based on destination IP address only-regardless of the destination port number.
 - **drop-unknown-mac** Drops packets when they contain unknown destination MAC addresses.
 - **queue-size *number*** Defines the number of proxied connection requests held in the proxied connection queue before the system starts rejecting new connection requests.
 - **source-threshold *number*** Specifies the number of SYN segments received per second from a single source IP address (regardless of the destination IP address and port number) before the NetScreen device begins dropping connection requests from that source.
 - **timeout *number*** Defines the maximum length of time before a half-completed connection is dropped from the queue. You can set it between 1 and 50 seconds.

- **syn-frag** Detects a SYN fragment attack, and drops any packet fragments used for the attack. A SYN fragment attack floods the target host with SYN packet fragments. The host caches these fragments, waiting for the remaining fragments to arrive so it can reassemble them. By flooding a server or host with connections that cannot be completed, the host's memory buffer eventually fills. No further connections are possible, and damage to the host's operating system can occur.
- **tcp-no-flag** Drops an illegal packet with missing or malformed flags field.
- **tear-drop** Blocks the Teardrop attack. Teardrop attacks occur when fragmented IP packets overlap and cause the host attempting to reassemble the packets to crash. The tear-drop option directs the NetScreen device to drop any packets that have such a discrepancy.
- **udp-flood threshold** *number* UDP flooding occurs when an attacker sends UDP packets to slow down the system to the point that it can no longer process valid connection requests. The **threshold** *number* parameter is the number of packets allowed per second to the same destination IP address/port pair. When the number of packets exceeds this value within any one-second period, the NetScreen device generates an alarm and drops subsequent packets for the remainder of that second. The valid range is from 1 to 1,000,000.
- **unknown-protocol** Discards all received IP frames with protocol numbers greater than 135. Such protocol numbers are undefined or reserved.
- **winnuke** Detects attacks on Windows NetBios communications, modifies the packet as necessary, and passes it on. (Each WinNuke attack triggers an attack log entry in the event alarm log.)

Example 1: The following command enables the **ip-spoofing** firewall service for the **trust** zone:

```
set zone trust screen ip-spoofing
```

The following command enables the **ip-spoofing** firewall service for the **untrust** zone, and instructs the device to drop any packet that has no source IP address, or that has a non-routable source IP address:

```
set zone untrust screen ip-spoofing drop-no-rpf-route
```

Example 2: The following command sets up a filter that scans HTTP packets for the **code-red** Code Red worm virus and drops such packets.

```
set zone untrust screen mal-url code-red
```

Example 3: The following commands block ActiveX and Java applets in HTTP traffic received on interfaces bound to the Untrust zone:

```
set zone untrust block-component activex
set zone untrust block-component java
```

Example 4: The following commands limit the number of sessions from any host in the Trust and Untrust zones to any single IP address to 80 sessions:

```
set zone trust screen limit-session destination-ip-based 80
set zone trust screen limit-session
set zone untrust screen limit-session destination-ip-based 80
set zone untrust screen limit-session
```

tcp-rst

```
set zone zone tcp-rst
unset zone zone tcp-rst
```

tcp-rst Directs the NetScreen device to send back the TCP reset packet when it receives non-sync packets.

vrouter

```
set zone zone vrouter
```

vrouter Binds the zone to a virtual router.

Creating Interfaces

Example: The following commands:

- Create a new Layer-2 zone named **L2-Marketing** with VLAN ID number 1
- Assign physical interface **ethernet7** to the zone
- Retrieve zone information:

```
set zone name L2-Marketing L2 1
set interface ethernet7 zone L2-Marketing
```


Example: The following commands:

- Create a new Layer-3 zone named **Ext_Dept**
- Bind the zone to the **untrust-vr** virtual router
- Enable **ip-spoofing** and **tear-drop** screening
- Bind interface **ethernet4** to the zone:

```
set zone name Ext_Dept
set zone Ext_Dept vrouter untrust-vr
set zone Ext_Dept screen ip-spoofing
set zone Ext_Dept screen tear-drop
set interface ethernet4 zone Ext_Dept
```


Interface Names

Most security zones exchange traffic with other zones (or with other devices) through physical interfaces or logical sub-interfaces. The interface names are as follows.

Aggregate interfaces	<ul style="list-style-type: none"> • aggraten An aggregate interface, which is a grouping of two physical interfaces. An aggregate interface provides interface redundancy, allowing load sharing and failover.
Ethernet interfaces	<ul style="list-style-type: none"> • ethernetn A physical ethernet interface, denoted by an interface port <i>n</i> and no slots. • ethernetn1/n2 A physical ethernet interface, denoted by an interface slot (<i>n1</i>) and a port (<i>n2</i>).
Function interfaces	<ul style="list-style-type: none"> • mgt An interface bound to the MGT zone. • ha ha1 ha2 The name of the dedicated HA port.
Layer-2 interface	<ul style="list-style-type: none"> • vlan1 The interface used for VPNs and management traffic while the NetScreen device is in Transparent mode.
Loopback interfaces	<ul style="list-style-type: none"> • loopback.n A logical interface that emulates a physical interface on the NetScreen device. A loopback interface is always in the up state as long as the device on which it resides is up.
Redundant interfaces	<ul style="list-style-type: none"> • redundantn1 A redundant interface, which is a grouping of physical interfaces (each denoted by <i>n1</i>). Redundant interfaces perform interface failover. • redundantn1.n2 A logical redundant sub-interface.
Sub-interfaces	<ul style="list-style-type: none"> • ethernetn1.n2 A logical sub-interface, denoted by an interface port (<i>n1</i>) with no slots. The <i>.n2</i> parameter identifies the logical interface. You create logical interfaces using the set interface command. • ethernetn1/n2.n3 A logical sub-interface, denoted by an interface slot (<i>n1</i>) and a port (<i>n2</i>). The <i>.n3</i> parameter identifies the logical interface. You create logical interfaces using the set interface command.
Tunnel interfaces	<ul style="list-style-type: none"> • tunnel.n A tunnel interface, used for VPN traffic.

Zone Names

Juniper Networks NetScreen devices use zones to host physical and logical interfaces, tunnels, and special-purpose items. Although ScreenOS has a number of default predefined zones, you can create new zones and configure them to meet the requirements of your organization. The names of ScreenOS security zones are as follows.

- Layer-2 security zones** Use Layer-2 security zones when the Juniper Networks NetScreen device operates in Transparent mode.
- **v1-trust** The V1-Trust zone, which hosts physical interfaces that communicate with trusted network space.
 - **v1-untrust** The V1-Untrust zone, which hosts physical interfaces that communicate with untrusted network space.
 - **v1-dmz** The DMZ zone, which hosts the DMZ physical interface.
 - **name** *name_str* A user-defined Layer-2 security zone. (You create such zones using the **set zone name** *name_str* **L2** command.)
- Layer-3 security zones** Use Layer-3 security zones when the Juniper Networks NetScreen device operates in NAT mode or Router mode.
- **trust** The Trust zone, which hosts physical interfaces (and logical sub-interfaces) that communicate with trusted network space.
 - **untrust** The Untrust zone, which hosts physical interfaces (and logical sub-interfaces) that communicate with untrusted network space.
 - **global** The Global zone, which serves as a storage area for mapped IP (MIP) and virtual IP (VIP) addresses. Because traffic going to these addresses is mapped to other addresses, the Global zone does not require an interface.
 - **dmz** The DMZ zone, which hosts the DMZ physical interface.
 - **name** *name_str* A user-defined Layer-2 security zone. (You create such zones using the **set zone name** *name_str* command.)

Tunnel zones

Use tunnel zones to set up VPN tunnels with other NetScreen security devices.

- **untrust-tun** The Untrust-Tun zone, which hosts VPN tunnels.
- **name** *name_str* A user-defined tunnel zone. You create such zones using the **set zone name** *name_str* **tunnel** command.

Function zones

Use function zones as described below.

- **null** The Null zone, which serves as temporary storage for any interfaces that are not currently bound to another zone.
- **self** The Self zone, which hosts the interface for remote management connections. For example, when you connect to the Juniper Networks NetScreen device via HTTP, SCS, or Telnet, you connect to the Self zone.
- **ha** The HA zone, which hosts the high-availability interfaces, HA1 and HA2.
- **mgt** The MGT zone, which hosts the out-of-band management interface, MGT.



Command Availability

Most CLI commands are available across all Juniper Networks NetScreen device platforms. However, some platforms do not support certain commands. The following table lists the CLI commands described in this manual, and shows which platforms do not support them.

Command	NS-5XP	NS-5XT	NS-5GT	NS-25	NS-50	NS-200	NS-500	NS-5000
address								
admin								
alarm								
alias								
arp								
attack	No							
attack-db	No							
audible-alarm	No	No	No	No	No	No		No
auth								
auth-server								
av	No	No					No	No
bgp	No							
chassis	No	No	No	No	No	No	No	
clock								
common-criteria								
config								
console								

Command	NS-5XP	NS-5XT	NS-5GT	NS-25	NS-50	NS-200	NS-500	NS-5000
counter								
delete								
di	No							
dip								
dns								
domain								
downgrade			No					
envar								
event								
exit								
failover	No			No	No	No	No	No
file								
firewall								
flow								
gate								
group								
group-expression								
hostname								
ike								
ike-cookie								
interface								
ip								
ip-classification	No	No		No	No	No		

Command	NS-5XP	NS-5XT	NS-5GT	NS-25	NS-50	NS-200	NS-500	NS-5000
ippool								
l2tp								
lcd	No	No	No	No	No	No		No
led	No	No	No					
license-key								
log								
mac								
mac-learn								
memory								
mip								
mirror	No	No	No	No	No	No	No	
modem	No			No	No	No	No	No
nrtip								
nsmgmt								
nsrp	No							
ntp								
os								
ospf	No							
performance								
ping								
pki								
policy								
port-mode	No			No	No	No	No	No

Command	NS-5XP	NS-5XT	NS-5GT	NS-25	NS-50	NS-200	NS-500	NS-5000
pppoe						No		No
proxy-id								
reset								
rip								
route								
sa								
sa-filter								
sa-statistics								
save								
scheduler								
scp								
service								
session								
sip								
snmp								
socket								
ssh								
ssl								
sys_clock	No	No	No	No	No		No	No
syslog								
system								
tech-support								
temperature-threshold	No	No	No	No	No	No		No

Command	NS-5XP	NS-5XT	NS-5GT	NS-25	NS-50	NS-200	NS-500	NS-5000
tftp								
timer								
trace-route								
traffic-shaping	No							No
url								
user								
user-group								
vip								
vpn								
vpn-group								
vpnmonitor								
vrouter								
vwire	No	No	No	No	No	No	No	
webauth								
webtrends								
xauth								
zone								

